VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV AUTOMATIZACE A INFORMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

NÁVRH A REALIZACE MODELU AUTOMATICKÉ PRAČKY

AUTOMATIC WASH MACHINE DESIGN AND REALIZATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JAN ŽELEZNÝ
AUTHOR

VEDOUCÍ PRÁCE — ING. TOMÁŠ MARADA, PH.D.
SUPERVISOR

BRNO 2012

ZADÁNÍ ZÁVĚREČNÉ PRÁCE

(na místo tohoto listu vložte originál a nebo kopii zadání Vaší práce)

ABSTRAKT

Bakalářská práce se zabývá návrhem a realizací modelu automatické pračky, v rámci souborů EDU-mod, pro podporu výuky předmětu Programovatelné automaty. Podrobně popisuje postup při výrobě od návrhu, přes realizaci, až po ověření funkčnosti modelu.
Základem desky je mikrokontroler Atmel ATmega16A. Program pro procesor byl vytvořen ve vývojovém prostředí Atmel AVR Studio, a následně nahrán pomocí programátoru UniProg-USB PK Design. Funkce modelu byly ověřeny řízením z PLC Siemens Simatic S7-200.

ABSTRACT

This bachelor‘s thesis is dealing with the design and realization of automatic wash machime, from EDU-mod models, as a support for Programmable logic controllers course. There is a detailed description of the manufacturing process, from design and realization, to verify the functionality of the model.

The basis of the board is microcontroller Atmel ATmega16A. Program for procesor was created in software Atmel AVR Studio, and then send to ATmega by programmer UniProg-USB PK Design. Functionality of the model were verified by controlling by PLC Siemens Simatic S7-200.

KLÍČOVÁ SLOVA

EDU-mod, PLC, ATmega16A, Siemens Simeatic S7-200, STEP7, DPS

KEYWORDS

EDU-mod, PLC, ATmega16A, Siemens Simeatic S7-200, STEP7, DPS

PROHLÁŠENÍ O ORIGINALITĚ

Prohlašuji, že jsem tuto práci vypracoval sám, bez cizí pomoci pouze na základě použité literatury

BIBLIOGRAFICKÁ CITACE

ŽELEZNÝ, J. *Návrh a realizace modelu automatické pračky*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2012. 70 s. Vedoucí bakalářské práce Ing. Tomáš Marada, Ph.D..

PODĚKOVÁNÍ

Tímto bych chtěl poděkovat vedoucímu mé bakalářské práce Ing. Tomáši Maradovi Ph.D., za vedení, a cenné rady, které mi byly poskytnuty v průběhu práce.

Obsah:

1 ÚVOD

Moderní výuka logických systémů a automatizace se v dnešní době neobejde bez používání programovatelných automatů (PLC), nebo jiných číslicových systémů.
Soubor modelů EDU-mod přispívá k této výuce díky simulaci různých technologických procesů, jako je například simulace funkcí jednotlivých procesů automatické pračky. Nevýhodou těchto modelů však mohou být jejich pořizovací náklady.

Cílem této práce je seznámit se s modelem automatické pračky EDU-mod. Zjistit principy funkcí a obvodů modelu a s využitím těchto znalostí realizovat obdobný model dostupnými prostředky laboratořeA1/731a.

2 MODELY SOUSTAV EDU-MOD

V první kapitole práce se blíže seznámíme s modely EDU-mod. Byly vyvinuty v rámci výukového systému EDUtec firmou TECO a.s.. Jsou využívány nejen při výuce na středních, vyšších i vysokých školách, ale i pro školení a výcvikové kurzy pořádané firmou TECO a.s..

Modely, schopné simulovat technologické procesy, jsou určeny pro praktickou výuku logických systémů, realizovatelných programovatelnými automaty (PLC), řídícími počítači, stavebnicemi logických obvodů (např. Dominoputer), atd.[1]

V Laboratoři A1/731a jsou pro studijní účely k dispozici modely křižovatky, hydraulické posuvné jednotky, mísící jednotky a automatické pračky. Tato práce se bude zabývat pouze modulem automatické pračky.

2.1 Dělení modelů

- Moduly EDU-mod řady 24V

Logické signály s úrovní 24 V ss umožňují univerzální použití pro libovolný typ PLC systému. Vstupní i výstupní signály jsou definovány proti společnému zápornému vodiči. Opačnou polaritu signálů je možno řešit přizpůsobovacími členy.[1]

- Moduly EDU-mod řady 5V

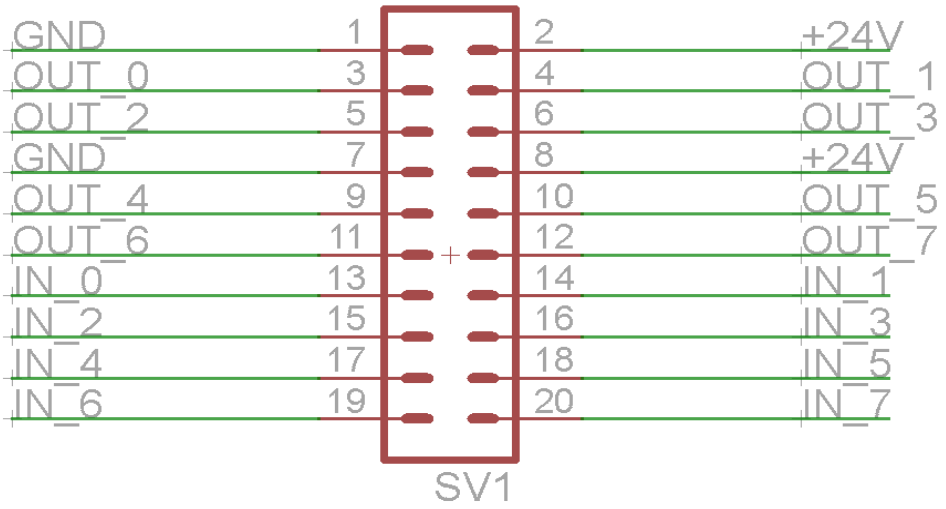
Logické signály s úrovní TTL dovolují spojení s logickými automaty realizovanými na bázi stavebnic číslicových IO, programovatelných logických polí (PLD), procesorových obvodů, atd.[1]

2.2 Vstupní a výstupní signály – propojení s PLC

Vstupní a výstupní signály EDU-mod jsou vedeny na 20 pólový konektor zajišťující propojení plochým kabelem s rozbočovacím modulem. Mechanické provedení je shodné s periferiemi EDUtec. Rozbočovací modul obsahuje 4 konektory Cannon 9 (2 vstupní, 2 výstupní), pro připojení max. 8 vstupních a 8 výstupních binárních signálů z/do libovolného systému. Chceme-li tedy použít zvolený modul, stačí jej kabelem připojit a do odpovídajících konektorů zapojit řídicí systém (např. EDUtec). [1]

V laboratoři A1/731a je místo rozbočovacího modulu použit přípravek: 20 žilový plochý kabel s potřebným 20 pinovým konektorem, který je zaveden přímo na porty PLC

Díky tomuto přípravku se výrazně zjednodušuje zapojení a instalace modulu se zrychlí. Schéma zapojení konektoru na obr. 1.



Obr.1 Schéma zapojení propojovacího konektoru

2.3 Model automatické pračky

Model automatické pračky patří k zajímavějším a sofistikovanějším modelům z hlediska programování a řízení. Je zde totiž možné reagovat na výstupy z modelu v podobě snímání hladiny a teploty.Zajímavou funkcí je možnost dvou rychlostí otáček pracího bubnu. Pro reálnější dojem zde po odpojení topení začne klesat teplota.

Pomocí tohoto modelu tedy můžeme simulovat automatickou pračku s většinou moderních dostupných funkcí.

Obr.2 Model automatické pračky

Vlastnosti modelu

- Model automatické pračky je řízen 6 binárními výstupy PLC jejich stav je zobrazen pomocí PLC. Vstupní i výstupní binární signály jsou na úrovni 24V logiky se společným záporným vodičem (GND).
- Dva výstupní signály ovládají směr otáčení bubnu. Pohyb bubnu je znázorněn LED diodami uspořádanými v kruhu formou "běžícího světla". Výstup OTÁČKY řídí rychlost otáčení bubnu (0 = praní, 1 = ždímání).
- Model simuluje napouštění, vypouštění a ohřev vody v prací vaně, a to včetně chladnutí. Napuštění řídí bit VODA, vypouštění bit ČERPADLO, a ohřev vody zase bit TOPENÍ.
- Model snímá výšku hladiny ve dvou úrovních na 50% a 100% naplnění. Tato informace je poslána na příslušné vstupy PLC a je zobrazena pomocí LED.
- Při ohřívání vody se model chová jako soustava 2. řádu, avšak časové konstanty jsou zkráceny tak, aby se při ladění aplikací nemuselo příliš dlouho čekat (ohřev plné vany na 90°C trvá cca 60s). Teplota vody je snímána ve 4 bodech (30, 40, 60 a 90°C).[1]

Inicializační stav

Při zapnutí napájení nebo po stisku tlačítka RESET se model nastaví do počátečního stavu. Jde o stav kdy je prací vana prázdná, teplota je na počáteční hodnotě a buben setrvává v klidu.

Chybová hlášení

Model generuje dva druhy chybových hlášení. Jde o chyby opravitelné a neopravitelné

- Opravitelná chyba nastane v případě, kdy řídící program vyšle pokyn k otáčení bubnu na obě strany ve stejnou chvíli. Tento stav je indikován blikáním LED ERR. Model není třeba restartovat, ale musí se změnit řídící program

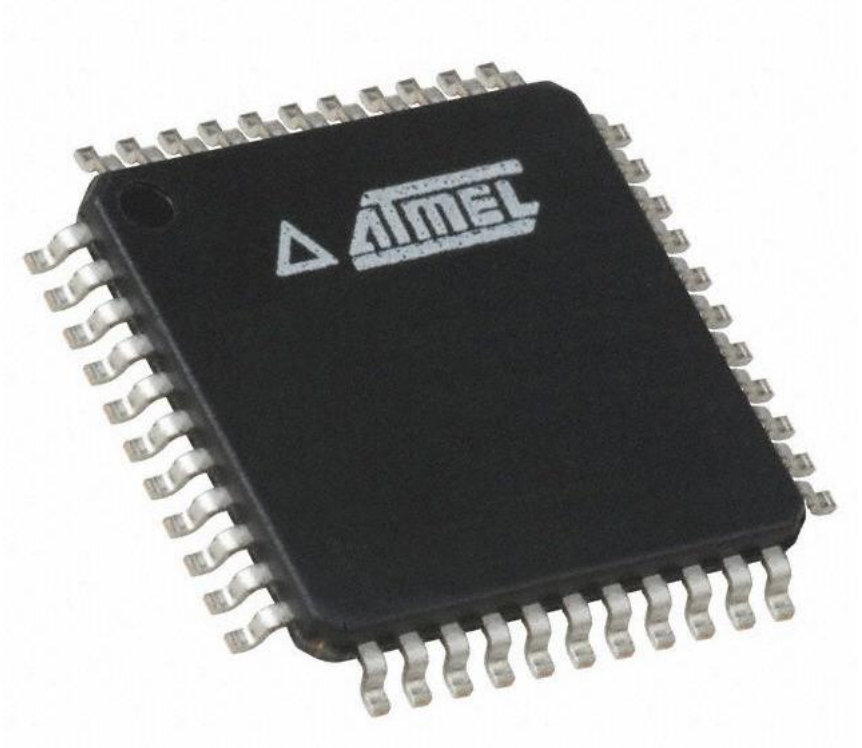
- Neopravitelná chyba je detekována, pokud voda v prací vaně stoupne nad 100%, nebo pokud teplota stoupne nad 90°C. V tomto stavu se rozsvítí LED ERR a pro odstranění chyby je nutné restartovat model tlačítkem RESET.

3 HARDWARE MODELU

Při navrhování modelu jsme vycházeli z původního zapojení modelu EDU-mod. Z obvodů jsme zjistili, že spodní deska plošných spojů (DPS), se u jednotlivých modelů liší počtem aktivních vstupních i výstupních členů komunikujících s PLC. Díky tomuto poznatku se stalo prioritou, aby spodní DPS byla navržena univerzálně, tak aby byla do budoucna použitelná jako základ dalších prací, které by měly za úkol návrh nového modelu.

Další změnou oproti původnímu návrhu měla být redukce počtu mikrokontrolerů. V návrhu od firmy TECO a.s. je umístěn jeden mikrokontroler Atmel AT89 na spodní DPS, který řídí 8 vstupních a 8 výstupních členů. Ty komunikují s PLC a zároveň řídí zobrazování procesů na horní DPS, kde vedou na LED, tlačítko RESET a dva programovatelné členy jsou vstupy do druhého mikrokontroleru. Procesor na horní DPS řídí pouze otáčení bubnu se dvěma vstupy a 8 výstupy v podobě diod uspořádaných do kruhu.

V novém návrhu se kladl důraz na nahrazení obou mikrokontrolérů jedním, a to Atmel ATmega16A, který můžeme vidět na obr. 3. Tento má dvakrát více vstupně výstupních (I/O) členů oproti AT89. Z 32 pinů jsme mohli použít 8 jako vstupních, 8 jako výstupních a dalších 16 jako volně programovatelných. Díky nim není potřeba dávat mikroprocesor i na horní desku. Další výhodou je menší velikost ATmegy čímž se ušetří místo na spodní DPS.



Obr. 3 Mikrokontroler Atmel ATmega16 – AU

Model tedy tvoří dvě DPS umístěny do krabičky WEB1001. Tu lze snadno přichytit k liště DIN.

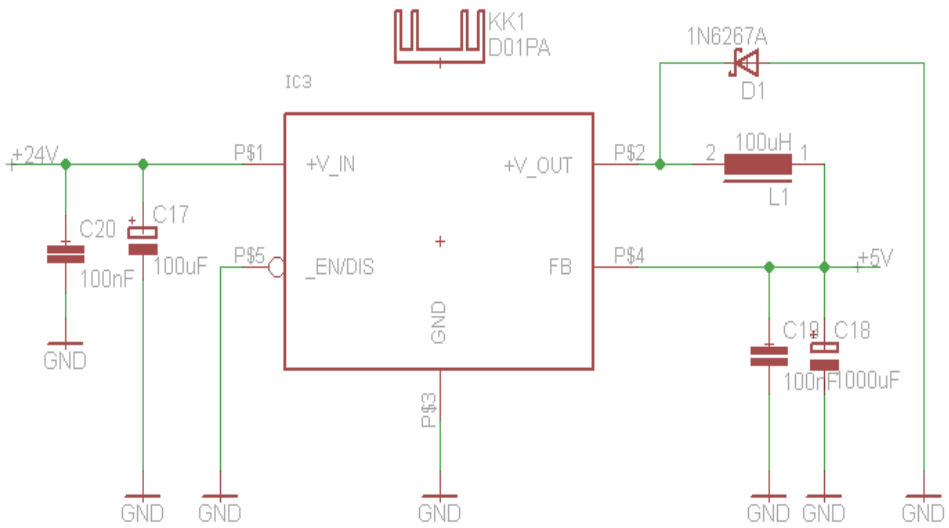
3.1 Stabilizátor LM2576

Řada LM2576 jsou monolitické integrované obvody, měniče napětí s funkcí step-down (snižující). Tyto spínací regulátory jsou schopné vydržet zatížení až do 3A, při maximálním vstupním napětí až 40V. Přes vysokou výstupní účinnost obvodu se přehřívá minimálně. Přesto jsme na jeho pouzdro TO-220 přimontovali hliníkový chladič DO3A.

Výhodou stabilizátoru je jeho jednoduché použití a, že přes jeho vlastnosti potřebuje k provozu minimum externích součástek. Stabilizátor jsme připojili podle návodu v dokumentaci, kde bylo doporučené zapojení součásti i s vlastnostmi externích součástek. Zapojení je na obr. 4.

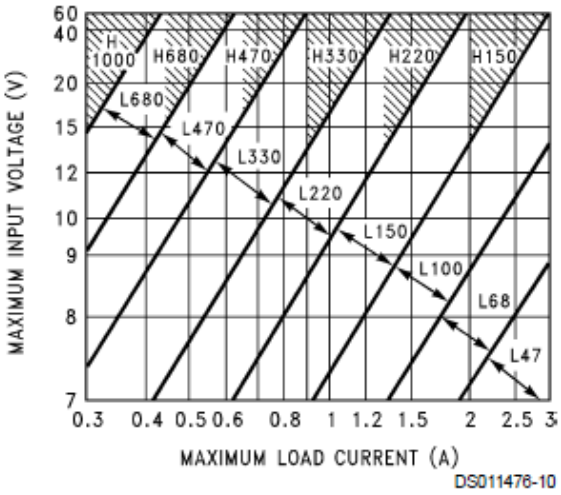
Vlastnosti

- 3,3V, 5V, 12V, 15V a nastavitelná verze
- Garantovaný proud do 3A
- Potřebuje 4 externí komponenty
- Frekvenční interní oscilátor s frekvencí 52kHz
- Účinnost 75%
- Využívá standardní indukční cívky
- Tepelná pojistka a proudový jistič



Obr. 4 Schéma zapojení Stabilizátoru LM2576

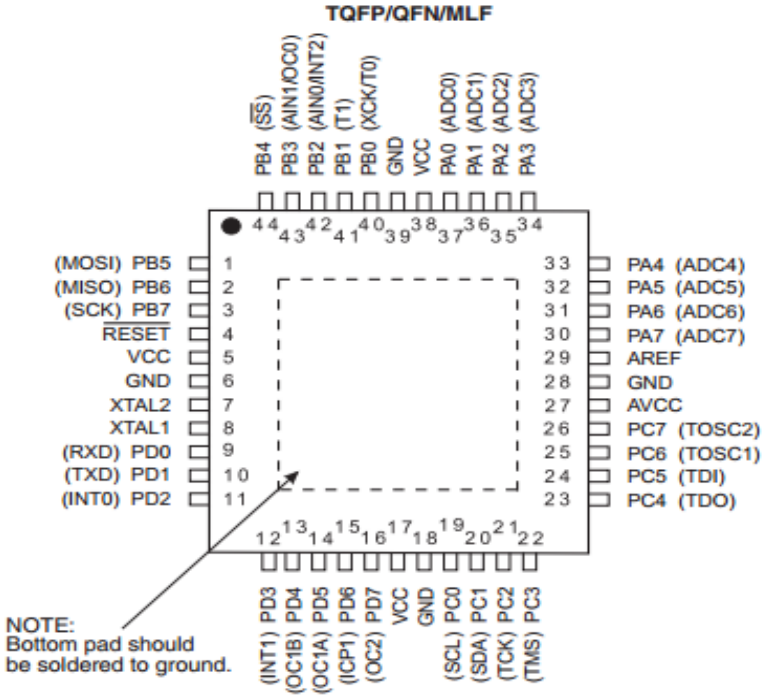
Velikost indukce cívky L1 (100uH) jsme volili z grafu dokumentace, podle velikostí proudu a napětí (I = 2,5 a U = 24V). Graf je na obr. 5.



Obr. 5 Graf volby indukce

3.2 Mikrokontroler Atmel ATmega16A

Jak bylo řečeno výše model je řízen SMD mikrokontrolerem ATmega. Je to nízkonapěťový mikroprocesor navržený tak, aby jeho AVR struktura vyhovovala pro užití vyšších programovacích jazyků. Na obr. 6 můžeme vidět jeho pin-out.



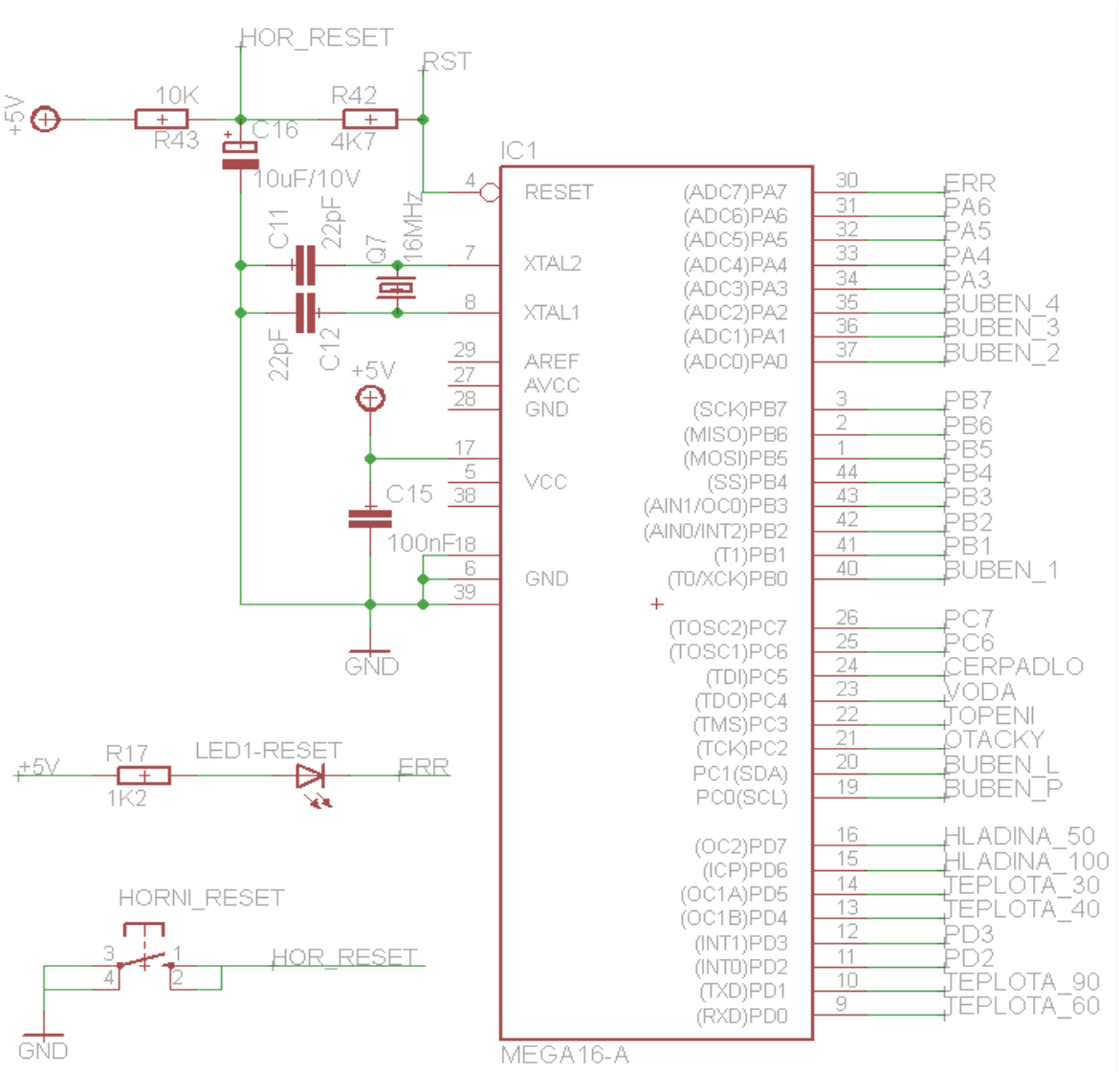
Obr. 6 Pin-out Mikrokontroleru ATmega16A

Vlastnosti

- vysoko výkonnostní, nízkonapěťový,AVR 8 bitový mikrokontroler
- rozšířená RISC architektura.
- paměťové segmenty
  - 16 kB programové FLASH paměť
  - 512 B EEPROM paměť
  - 1 kB interní operační paměť SRAM
- JTAG, ISP rozhraní
- periferní funkce
  - 2 x 8 bitový čítač/časovač
  - 1 x 16 bitový čítač/časovač
  - obvod reálného času s odděleným oscilátorem
  - sériové rozhraní USART
  - master/slave sériové rozhraní SPI
  - watchdog se separátním oscilátorem
  - AD převodník (8 kanálové, 10 bitové rozlišení)
  - 4 kanály PWM
- 32 programovatelných I/O portů
- Provozní napětí 2,7 – 5,5 V

3.3 Zapojení ATmega16A do bloku

Pro správnost funkce je ATmega zapojena v bloku s několika základními pasivními součástkami. Zapojení řídícího bloku můžete vidět na obr. 7.



Obr. 7 Zapojení ATmega do řídícího bloku

Mikrokontrolér je napájen přivedeným napětím na jeden z pinů VCC. Jde o +5V přiváděných ze stabilizátoru LM2576. Zem je přivedena na piny GND pomocí vylití větších ploch mědí.

Restartování modelu je zajištěno v obvodu, který je připojen k pinu RESET. Tento obvod se skládá z dvou rezistorů (R42 a R43) s odpory 10kΩ a 4,7kΩ a z kondenzátoru (C16) s kapacitou 10uF. Takto zajišťuje hladký, a zpožděný náběh resetovacího signálu. Kromě programování je využit i na restartování všech funkcí modelu. Restartování modelu je řízeno tlačítkem HOR_RESET umístěným na horní DPS.

ATmega je rozšířená krystalem o velikosti 16 MHz připojeným na piny XTAL1 a XTAL2, které jsou k tomu určené. Funguje zde jako externí zdroj pracovního kmitočtu. Zapojený je v obvodu s párem totožných kondenzátorů (C11 a C12) s kapacitou 22 pF, zvolených podle návodu z dokumentace výrobce.

Piny Portu PD jsou nastaveny jako výstupní členy ATmegy, proto jsou propojeny na vstupy PLC. Na výstupy automatu jsou připojeny piny portu PC, které jsou nastaveny jako výstupní členy mikrokontroleru. Zbylé dva porty PA a PB jsou zavedeny přímo na horní DPS jako pomocné programovací piny. V našem případě jsou čtyři z nich využity pro realizaci otáčení bubnu pračky formou "běžícího světla".

Z pinů portu PB jsou tři členy použity zároveň jako piny programovacího konektoru. Díky tomuto zapojení lze ukládat program z PC do mikrokontroleru. Jde o bity PB5, PB6 a PB7. Spolu s nimi je na programovací konektor přiveden i signál reset RST.

Na bitu PB5 je umístěn MOSI (Master Out/Slave In). Datový výstup řídícího členu a vstup řízeného členu sériového periferního rozhraní (SPI). Na bitu PB6 je MISO (Master Input/Slave Out), který má vlastnosti invertované oproti předchozímu, tzn.: vstup řídícího členu a výstup řízeného.

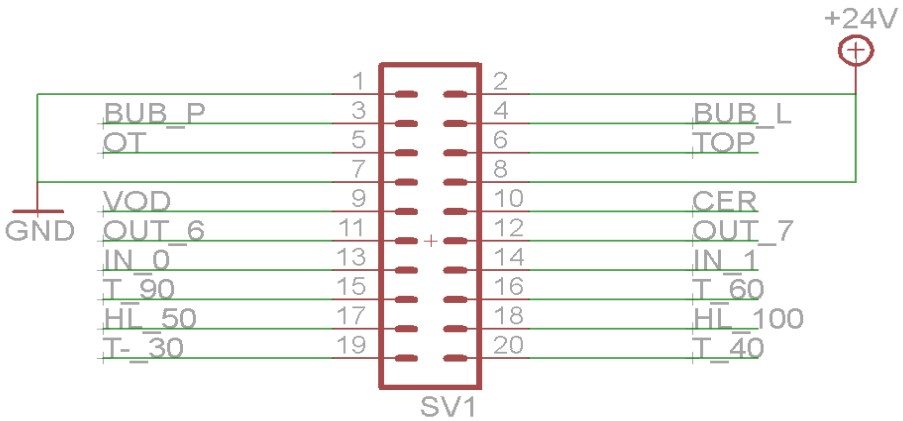
Bit PB7 je SCK (SPI bus Serial Clock), na tento bit se generuje hodinový signál, pomocí kterého se řídí komunikace Master/Slave.

3.4 Komunikace modelu s PLC

Základem fungující komunikace mezi PLC a modelem musí být navržení vstupních a výstupních obvodů tak, aby hodnoty proudu i napětí byly v mezích bezpečnosti a zároveň dostatečné pro logiku ATmegy a PLC. Obvody bylo tedy nutno navrhnout pro komunikaci komunikaci z PLC do mikroprocesoru i naopak.

Vycházelo se z původního návrhu DPS EDU-mod a ze specifikací I/O portů z dokumentace mikrokontroleru ATmega16A a programovatelného automatu firmy Siemens Simatic S7-224XP. Počítali jsme s rozdílnými napěťovými úrovněmi automatu a mikrokontroleru.

Kontrolu navrženého zapojení jsme prováděli realizací zapojení obvodů v nepájivém poli. Fyzicky byl pro propojení DPS s PLC použit 20 pinový konektor PSL20. Zapojení vstupů a výstupů konektoru modelu je vidět na obr. 8.



Obr. 8 Zapojení Konektoru pro komunikaci modelu

3.4.1 Logika PLC

Specifikaci digitálních vstupů a výstupů PLC jsme čerpali z dokumentace Siemens Simatic-S7-200 s CPU 224XP. Napěťová logika a její úrovně jsou popsány v tab. 1 a 2. U logické 1 je vždy uvedena dolní (min) hranice, kterou automat potřebuje k rozpoznání. U logické 0 je to naopak horní (max) hranice.

Specifikace digitálních výstupů	
Logická 1	L+ minus 0,4 V při max. proudu
Logická 0	0,1 V DC při zatížení 10 kΩ

Tab. 1 Specifikace digitálních výstupů PLC

Specifikace digitálních vstupů	
Logická 1	15 VDC při 2,5 mA
Logická 0	5 VDC při 1 mA

Tab. 2 Specifikace digitálních vstupů PLC

3.4.2 Logika mikrokontroleru

Nejčastěji se jako logika mikroprocesorů využívá tzv. TTL (Transistor Transistor Logic). Jde o standart využívaný pro implementaci digitálních integrovaných obvodů, které vycházejí z technologie bipolárních křemíkových tranzistorů. TTL obvody využívají napájecí napětí 5 V. Úrovně napětí logických proměnných jsou specifikovány v tab. 3.[4]

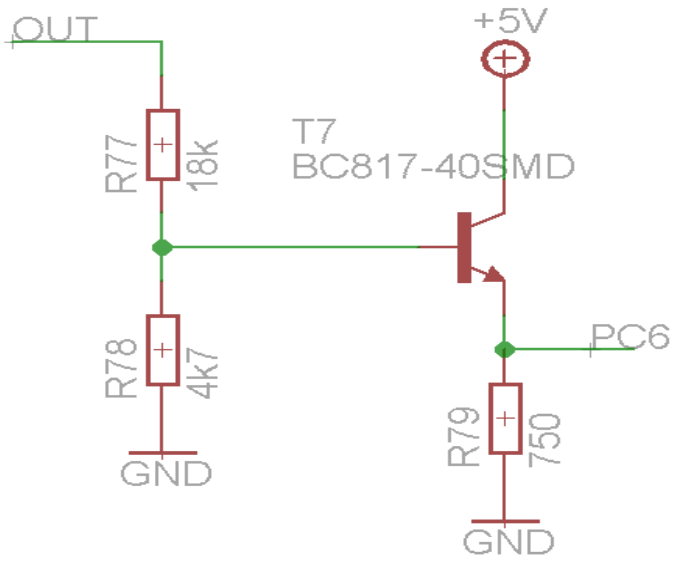
Specifikace úrovní TTL logiky	
Logická 1	15 VDC při 2,5 mA
Logická 0	5 VDC přii 1 mA

Tab. 3 Specifikace úrovní TTL logiky

3.5 Komunikace PLC → Mikrokontroler

Komunikace ve směru z automatu do mikrokontroleru musí řešit rozdílnou logiku obou zařízení. Jak bylo uvedeno v předchozím textu mikrokontrolery ATmega fungují na základě TTL, 5V logiky. PLC automaty využívají 24V logiku. Obvod, který jsme navrhli, obsahuje bipolární NPN tranzistor který má za úkol spínat vstupy ATmegy napětím +5V v případě že z automatu přijde logická 1.

Port PC ATmegy je připojen k emitoru. Pokud tedy přijde z PLC (OUT) logická 1 na bázi, tranzistor sepne napětí +5 V, které je připojené ke kolektoru, na emitor odkud jde přímo do mikrokontroleru. Ten přijme logickou 1. Pokud je vstup PLC v logické 0, tranzistor se nesepne a pin ATmegy zůstane připojen k zemi (GND), a tak bude na logické 0. Signál z tranzistoru jde, kromě do ATmegy, také přímo na diodu na horní DPS zobrazující určitou funkci přiřazenou k danému pinu mikrokontroleru. Zapojení obvodu je na obr. 9.



Obr. 9 Schéma zapojení komunikace PLC → Mikrokontroler

Při výpočtu výstupního napětí z tranzistoru jsme počítali s odpory R77 (R_1) a R78 (R_2), abychom se vlezli do minimální hodnoty napětí pro otevření kolektoru (5V).

[1]

$$U_{VÝST} = U_{VSTUP} \frac{R_2}{R_1 + R_2}$$

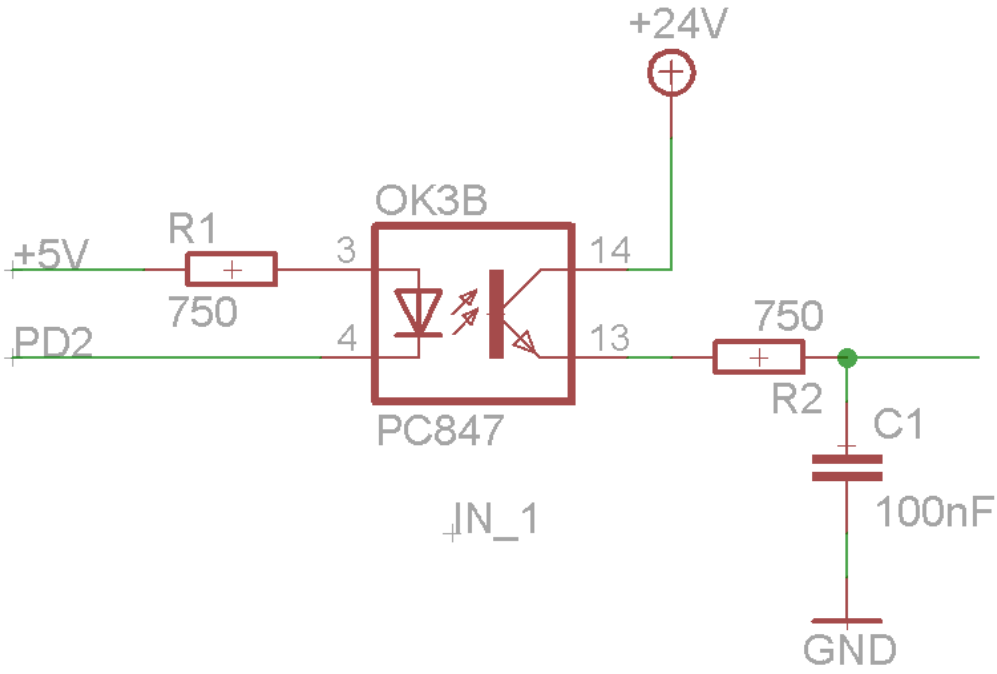
$$U_{VYST} = 24V \frac{4700\Omega}{1800\Omega + 4700\Omega} = 4{,}969V$$

3.6 Komunikace Mikrokontroler → PLC

V tomto směru komunikace jsme využili optočlenu s fototranzistorem, viz obr. 10. Optočleny jsou prvky, kde do jednoho pouzdra jsou vloženy dvě polovodičové součástky. V našem případě LED a fototranzistor. Slouží ke galvanickému oddělení dvou obvodů.

Vstupní obvod přivede proud na LED diodu a ta se rozsvítí. Otevření fototranzistoru

závisí na množství proudu protékajícím LED diodou. Závislost a vlastnosti použitého optočlenu PC847 jsou uvedeny v dokumentaci výrobce.
Tento obvod tedy opět řeší problém s rozdílem napěťových logik PLC a ATmegy.



Obr. 10 Schéma zapojení komunikace Mikrokontroler → PLC

Vstupy ATmegy jsou vedeny z portu PD na katodu LED diody optočlenu. Z druhé strany je proud ze zdroje omezen odporem R1. Vypočítáme proud dostatečný pro sepnutí opočlenu. Potřebné hodnoty jsme vyčetli z dokumentace produktu. Musíme počítat s úbytkem napětí na diodě.

[2] $$I = \frac{U_{VSTUP} - U_{LED}}{R1} = \frac{5V - 1{,}2V}{750\Omega} = 0{,}005A$$

Pokud je brána ATmegy nastavena na logickou 1 LED dioda se nerozsvítí, fototranzistor se neotevře a vstup automatu (resp. vstup ATmegy) bude propojen s GND. PLC v tomto stavu rozpozná logickou 0

V případě, kdy bude brána ATmegy nastavena na logickou 0, rozsvícená LED dioda otevře fototranzistor a na výstup PLC půjde proud +24 V omezený odporem R2. PLC detekuje logickou 1. Proud přicházející na výstupy PLC se spočítal ze zapojení následovně:

[3] $$I = \frac{U}{R} = \frac{24V}{750\Omega} = 0{,}32A$$

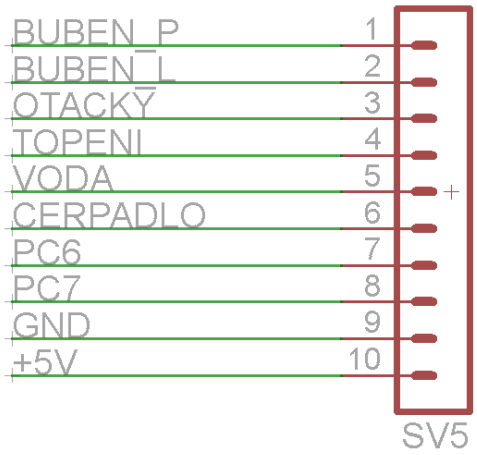
Toto zapojení způsobuje obrácení logiky, které je ošetřeno softwarově.

3.7 Zobrazování pomocí LED

Zobrazování stavů a simulací procesů je navrženo na horní DPS. Realizováno je pomocí několika druhů obvodů s LED diodami. Obvody se liší podle logiky vstupního nebo výstupního členu připojeného k diodě. Vstupní členy jsou realizovány na invertovanou logiku z důvodů zapojení komunikace ATmegy s PLC. Výstupní mají standardní logiku.

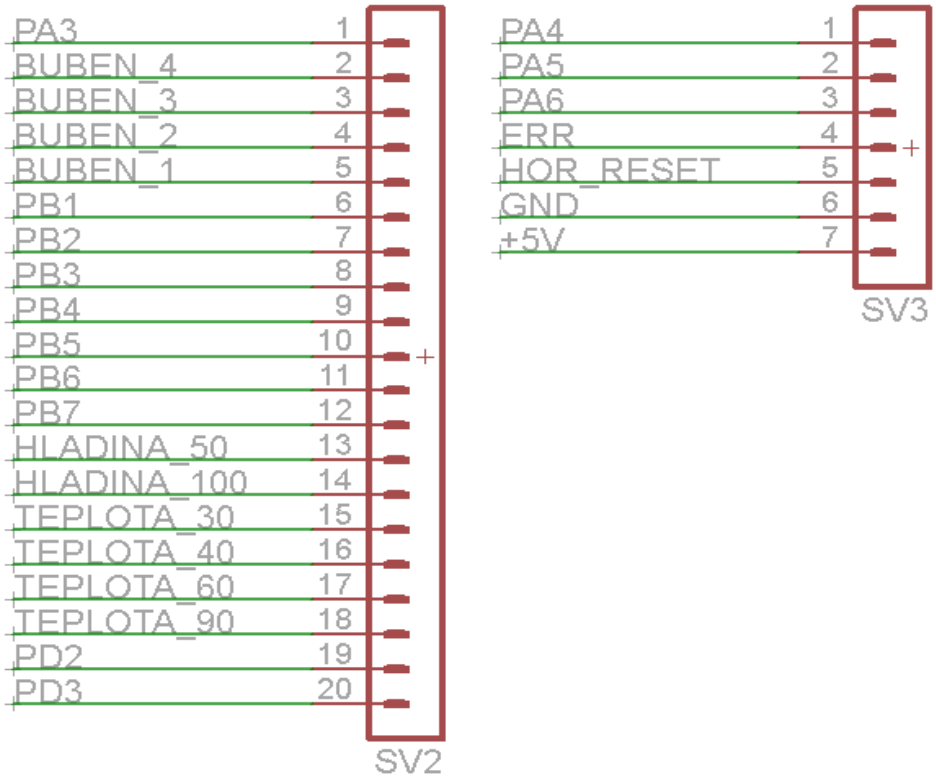
K propojení obou desek slouží oboustranné kolíky na spodní DPS, ke kterým jsou připojeny vstupy a výstupy ATmegy. Na horní DPS jsou použity dutinkové lišty.

K 10 propojovacím kolíkům je připojeno všech 8 výstupních členů. Výstupní port ATmegy je port PC. Poslední dva piny kolíků (9 a 10) jsou vyhrazeny pro +5V a GND. Zapojení je na obr. 11.



Obr. 11 Schéma zapojení výstupů na oboustranné kolíky

Vstupní členy mikrokontroleru jsou zapojeny na 20 pinové a 7 pinové oboustranné kolíky. Vstupy z portu PD, jejichž stavy se mohou zobrazovat v PLC, jsou zapojeny spolu s porty PA a PB, které slouží jako pomocné programovatelné. Opět jsou dva krajní piny vyhrazeny pro +5 v a GND. Dále jeden pro tlačítko Resetu HOR_RES a LED ERR zobrazující chybové stavy. Schéma zapojení na obr. 12.



Obr. 12 Schéma zapojení vstupů na oboustranné kolíky

3.7.1 Indikace vstupních členů

Vstupní členy PLC modelu automatické pračky simulují napouštění vany a ohřev vody. Informace o stavu je odváděna do ATmegy i do PLC. Proto musí být připojeny na piny portu PD.

Tento port komunikuje s PLC přes optočleny, kde se obrací logika. Na to se muselo myslet při navrhování obvodu, aby byla logika převrácena i zde. Schéma na obr. 13.

Obr. 13. Schéma zapojení vstupních členů

K anodě je připojen proud omezený odporem R1 na požadovanou hodnotu. Na katodu vede vstup ATmegy.

Pokud je brána mikrokontroleru nastavena na logickou 1, tak se LED nerozsvítí. Pokud je však nastavená na logickou 0, dioda se rozsvítí. Tento obvod dodržujeme stejnou obrácenou logiku jako u zapojení optočlenů. To se řeší softwarově.

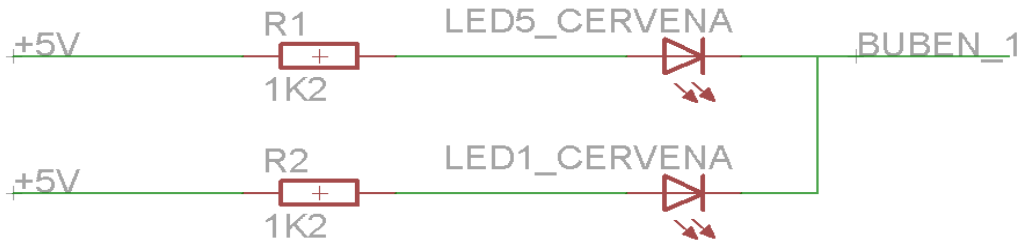
Potřebný proud na LED jsme spočítali pomocí Ohmova zákona. Při výpočtu se muselo myslet na úbytek napětí na LED U_{LED} Tento výpočet platí pro všechny diody použité na indikaci stavů modelu.

[4] $$I = \frac{U_{VSTUP} - U_{LED}}{R1} = \frac{5V - 1{,}2V}{1200\Omega} = 0{,}003A$$

3.7.2 Indikace bubnu pračky

Původní zapojení LED diod, které simulovaly otáčení bubnu automatické pračky, bylo realizováno druhým mikrokontrolerem s dvěma vstupy a 8 výstupy na každém byla připojena LED dioda. Těchto 8 diod bylo uspořádáno do kruhu. Nastavené byly programově formou "běžícího světla" tak, že vždy dvě diody naproti sobě svítily při simulaci otáčení.

V našem modelu jsme místo dalšího mikrokontroleru mohli využít volné programovací piny portů PA a PB. Další změnou bylo zredukování počtu pinů. Místo aby se dvě protichůdné svítící diody ošetřily softwarově, tak jsme je fyzicky propojily. Tím se nám zmenšil počet nutných pinů z 8 na 4. Schéma zapojení je na obr. 14.



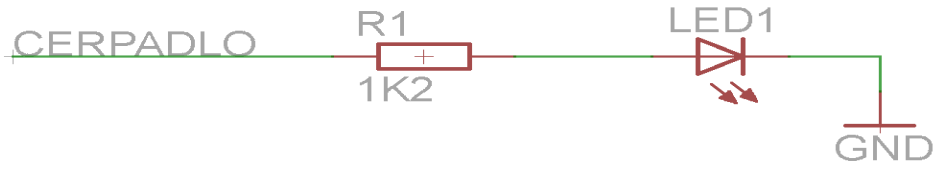
Obr. 14 Schéma zapojení bubnu pračky

Princip zapojení obvodu je stejný jako u vstupů ATmegy. Obrácená logika je ošetřena softwarově

3.7.3 Indikace výstupních členů

Na LED diody, které zobrazují stav výstupů PLC, je přiveden signál přímo z PLC přes tranzistor. Z tranzistoru vede zároveň na port ATmegy PC.

Navržený obvod je jiný než u vstupních členů, protože obvod tranzistoru neobrací logiku. Schéma obvodu na obr. 15.



Obr. 15 Schéma zapojení výstupních členů

Pokud se vstup PLC nastaví do logické 1, přijde na anodu LED diody proud snížený odporem R1. Dioda se rozsvítí, protože je zapojena v propustném směru, na katodě je zapojeno GND. Pokud se nastaví do logické 0, na anodu nepřijde proud a ta se nerozsvítí.

4 VÝROBA MODELU

Při realizaci modelu jsme museli postupovat po jednotlivých krocích. Prvním krokem bylo seznámení se s funkcemi a obvodu modelu automatické pračky EDU-mod. S využitím těchto poznatků jsme začali navrhovat obvody nového modelu.

Dále jsme se pustili do tvorby samotné DPS a jejího osazení součástkami. Po dokončení výroby jsme model proměřili a odzkoušeli, zda všechny obvody pracují korektně.

4.1 Eagle

K návrhu obvodů a DPS jsme používali program Eagle. Tento editor plošných spojů je produktem firmy CadSoft. Samotná název EAGLE je zkratkou názvu produktu „Easily Applicable Grafphical Layout Editor“, v překladu tedy: Snadno použitelný „Grafický Projektový Editor“.

Tento software používá jedno uživatelské prostředí, ze kterého ovládá tři hlavní moduly, ze kterých se program skládá.

Základní části programu:

- Editor schémat
- Editor spojů
- Autorouter

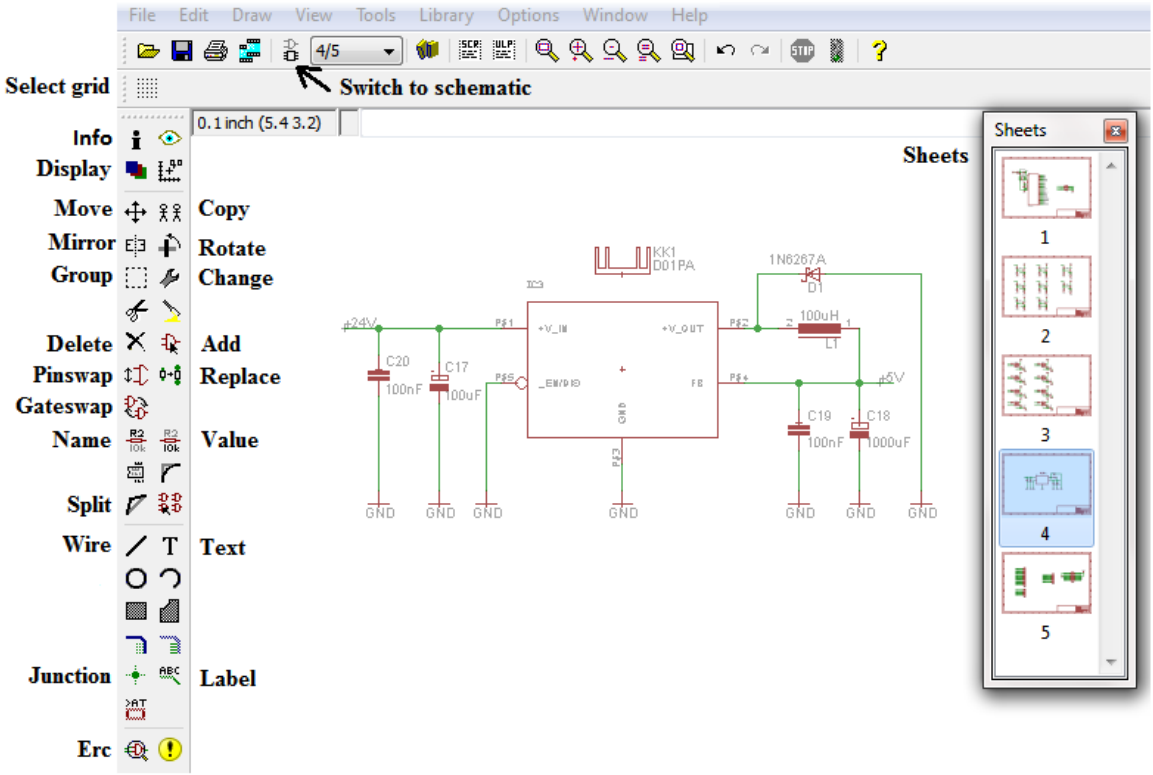
V ovládacím panelu lze jednoduše přepínat mezi těmito moduly bez nutnosti přizpůsobování netlistů mezi schématy a plošnými spoji.

Knihovna obsažená v programu obsahuje většinu základních a SMD součástek. Je tu také možnost editace, nebo vytvoření vlastní součásti.

4.1.1 Editor schémat

V tomto nástroji se vytvářejí a editují el. obvody. Na začátku bylo nutné nastavit rast na 2,54 mm (100 mil). Jinak by mohl nastat problém s propojením vývodů součástek. Při vkládání z knihovny jsou součástky automaticky označovány pro export seznamu součástek.

Na obr. 16 je náhled Editoru schémat s popisem využívaných funkcí.



Obr. 16 Editor schémat

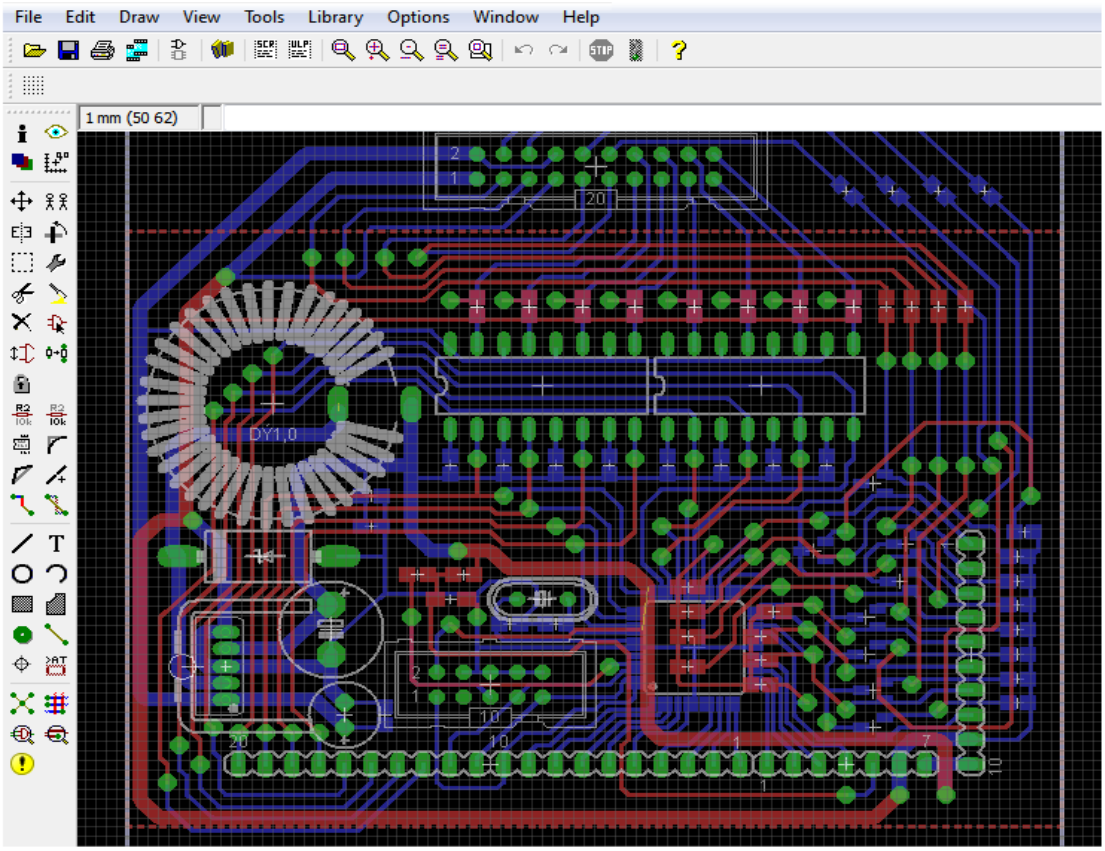
Význam a funkce jednotlivých tlačítek:

- **Switch to Board** - přepnutí do editoru plošných spojů. Usnadňuje práci při úpravách DPS
- **Grid** - nastavení mřížky (rastru)
- **Info** - při kliknutí na objekt se zobrazí veškeré dostupné informace (typ součástky, název, hodnota, pouzdro, knihovna)
- **Display** - nastavení viditelnosti vrstev
- **Move** - uchopení a přesunování součástek a drátů, pravým ladítkem myši hýbeme skupinou součástek
- **Copy** - kopírování součástek
- **Mirror** - zrcadlení součástek
- **Rotate** - rotuje součásti o 90°, také při stisku pravého tl. Myši
- **Group** - označení skupiny objektů, se kterými lze dále pracovat
- **Change** - funkce, která mění vlastnosti, pouzdra a mnohé další vlastnosti objektů
- **Delete** - smazání součásti, čáry
- **Add** - funkce na vybrání a vložení nové součásti
- **Replace** - při kliknutí na součást umožňuje její výměnu za jinou

- **Pinswap** - Prohození shodných vývodů součásti (pokud to součást umožňuje)
- **Gateswap** - Prohození např. hradel (pokud to součást umožňuje)
- **Name** - pojmenování objektu
- **Value** - hodnota součástky
- **Split** - zalomení čáry
- **Wire** - nakreslení čáry nebo propojení součástek
- **Label** - zobrazí název vodiče
- **Erc** - kontrola el. pravidel, zobrazí chyby a varování programu, zde je důležité aby nám hlásilo: „ Board and Schematic are consistent“, tj. jsou vzájemně propojené a změny se ukládají i v druhém modulu
- **Sheets** - snadné přepínání jednotlivými schématy

4.1.2 Editor spojů

V editoru spojů se navrhuje rozmístění součástek na DPS. Nejdříve určíme velikost desky a poté umísťujeme součástky podle vlastního uvážení. Editor používá pouzdra součástek pro lepší představu o rozmístění. Dalším krokem je návrh cest pomocí funkce Route. Při návrhu oboustranné desky se využívá možnosti nastavování vrstev. Přechody mezi deskami jsou realizovány prokovy funkcí Via. Editor schémat je na obr. 17.



Obr. 17 Editor spojů

4.1.3 Autorouter

Autorouter se nachází v editoru spojů. Pomocí této funkce lze automaticky propojit cesty mezi součástmi na DPS. Je potřebné nastavit šířky cest, izolačních mezer a dalších parametrů, před použitím funkce.

Autorouter je využitelný spíše u méně složitých návrhů DPS.

4.1.4 Knihovny

V této části Eaglu lze editovat, nebo vytvářet nové součásti. Knihovna prvky se skládá z 3 základních celků: Device, Package a Symbol.

Při tvorbě nového objektu se nejdříve vytvoří symbol s potřebnými piny. Druhým krokem je vytvoření pouzdra pro editor spojů. Dokončením součástky je propojení pájivých polí pouzdra s piny schematické značky.

4.2 Návrh DPS

Po seznámení se s programem Eagle jsme začali navrhovat obě DPS. Rastr Grid je nutné nastavit na hodnotu 25 mil (0,635 mm). Bez tohoto nastavení rastru by mohlo dojít ke špatnému zapojení. Rastr Grid byl takto nastaven po celou dobu práce, při navrhování DPS i při tvorbě nové součástky.

Nejdříve jsme začali tvořit schémata obvodů v Editoru schémat. Při vyhledávání součástek v knihovnách jsme zjistili absenci stabilizátoru LM2576T-5. Tuto součástku jsme tedy museli vytvořit v editaci knihoven. Podle dokumentace výrobce jsme navrhli schematickou značku stabilizátoru, kterou jsme propojili se standardním pouzdrem TO-220, ve které se stabilizátor běžně vyrábí. Návrh obvodu zapojení jsme vzali technické dokumentace součástky.

V Editoru spojů jsme navrhovali rozložení a cesty na DPS. Návrh vznikal s ohledem na výrobní podmínky laboratoře A1/731a. Nejdříve jsme vymezili prostor DPS rámečkem. Pro spodní DPS byla velikost 93x90 mm, pro horní 90x63 mm. Tyto rozměry byly převzaty z původního návrhu EDU-mod tak, aby se spojené DPS vešly do krabičky WEB1001.

Dále jsme nastavili šířku cest na doporučených 16 mil. Ve dvou případech jsme šířku cesty zvětšili. Šlo o cesty signálu přicházející přímo z PLC, Zem (GND) a napětí (+24 V). U těchto cest byla šířka zvolena na 56 mil s ohledem na důležitost těchto cest. Dále jsme určili minimální mezeru mezi cestami na 10 mil.

S ohledem na výrobní možnosti jsme museli také navrhnout optimální velikost pájecích plošek a prokovů. Jako tvar pájecí plochy prokovů byl zvolen osmiúhelník a velikost byla nastavena jednotně na 70 mil. U pájecích plošek součástí nebyl nastaven jednotný tvar s ohledem na možnosti návrhu. Velikost se pohybovala od 50 mil (Konektory), až po 100 mil (Kondenzátor, a Zenerova dioda). Velikost děr pro vrtání je jednotná, zvolena na 10 mil.

4.3 Výroba DPS

Existuje několik způsobů, jak vyrobit DPS. Nejdříve jsme museli posoudit, který z možných způsobů výrobu máme zvolit. Kritérii při této volbě byla dostupnost vybavení v laboratoři A1/731a, nenáročnost procesu a hlavně složitost navrhované desky. Podle těchto podmínek jsme zvolili metodu leptání fotocestou. Tak můžeme v našich výrobních podmínkách dosáhnout přijatelných výsledků.

Materiálem polotovaru DPS byla zvolena pertinaxová deska s fotocitlivou povrchovou vrstvou mědi. Jde o izolační materiál sklolaminát, který je tvrzený fenolformaldehydovou pryskyřicí. U nás se dodává pod obchodním názvem Cuprexit.[7]

Použité nástroje a vybavení při výrobě DPS:

- Předloha na osvitové fólii
- Nůžky, průhledná izolepa
- Osvitová jednotka LED
- Nádobky na leptání a vyvolání
- Vývojka (1,5 roztok NaOH)
- Leptací roztok (FeCl3)
- Aceton
- Pákové nůžky
- Stolní vrtačka na DPS
- Voděvzdorný smirkový papír
- Pájecí lak (sprej)
- Pájecí stanice, letovací drát

4.3.1 Osvitová předloha

Tvorba osvitové předlohy byla prvním krokem ve výrobě DPS. Princip výroby je v tom, že vrstva fotoemulze na desce se ve vývojce rozpustí, pokud byla osvětlena UV světlem, kdežto části neosvětlené zůstanou neporušeny.

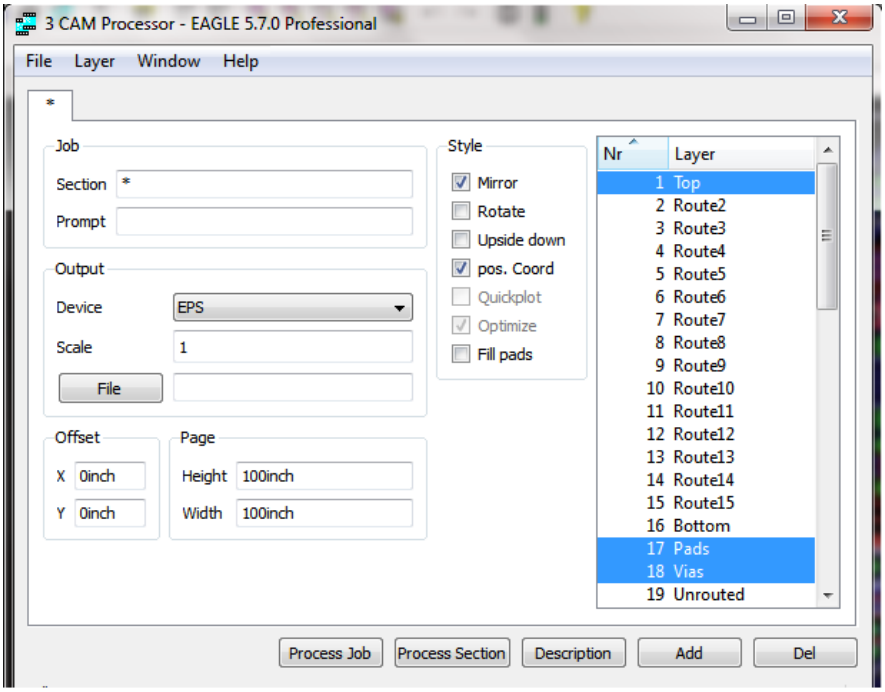
Musel se vybrat ideální postup, při kterém se vytvoří kvalitní předloha pro vícenásobné použití. Proto odpadají jednodušší možnosti, jako je vytisknutí laserovou tiskárnou na pauzovací papír, kde je problém s krytím větších černých ploch a kontrastem. Předloha by měla ideálně být vytištěna zrcadlově, aby mohla být přiložena potištěnou stranou přímo k desce. Tak bude kresba ostřejší. Samozřejmostí musí být vysoký kontrast v UV spektru, ostrost, rozlišení a přesnost předlohy.

Za zvolených podmínek Jsme zvolili předlohu osvícenou na film. Tento typ nelze pořídit doma. Naštěstí existuje několik tiskařských firem poskytující službu tisku osvitové předlohy pro DPS. Tyto firmy pak poskytují návod na předtiskovou přípravu předlohy. Je doporučené řídit se tímto postupem, aby nedošlo k zdlouhavým opravám souboru určeného k tisku.

Doporučené vlastnosti souboru:

- Kompozitní soubory formátu PDF
- Barevné schéma v režimu CMYK

Podle návodu na stránkách grafického studia jsme postupovali při vytváření PDF souboru s předlohou. Jako výstup z programu jsme zvolili funkci Eaglu CAM Processor, který se používá na všechny standardní výstupy z programu (tiskárna, plotter, obrázek). Nastavení CAM Processoru je na obr. 18.

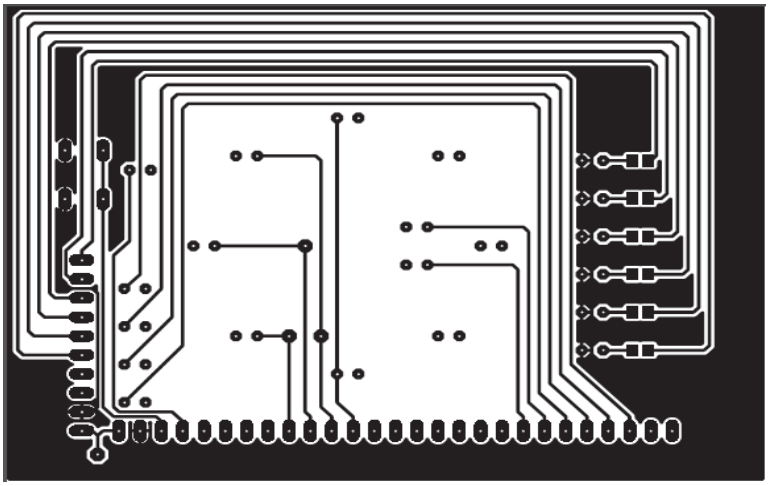


Obr. 18 Nastavení CAM Processoru

V tomto okně nastavíme vrstvy, které chceme přenést do obrázku a případné zrcadlení. Dále druh výstupního souboru nastavíme na EPS, který je otevíratelný v Adobe Illustratoru. Tam se posléze upraví barevné schéma na CMYK.

Podle doporučení firmy se má soubor konfigurovat v programu Acrobat Distiller. Ten se musí nastavit podle jejich návodu a na výstupu programu se vytvoří soubor PDF s požadovanými vlastnostmi pro tisk. Při práci v Adobe Illustratoru jsme zjistili, že výstup PDF z tohoto program lze také použít při tisku, aniž by se musel použít Distiler, což značně zjednodušuje přípravu tiskového souboru.

Při tisku obou stran jedné DPS je potřeba stranově zrcadlit horní stranu DPS (Top), a zároveň nechat spodní stranu (Bottom) v nepřevráceném stavu. Po vytisknutí jsme v Illustratoru obrázky umístili do jednoho souboru tak, aby byl po okrajích a mezi obrázky zachován dostatečný prostor pro zpracování k osvitu.

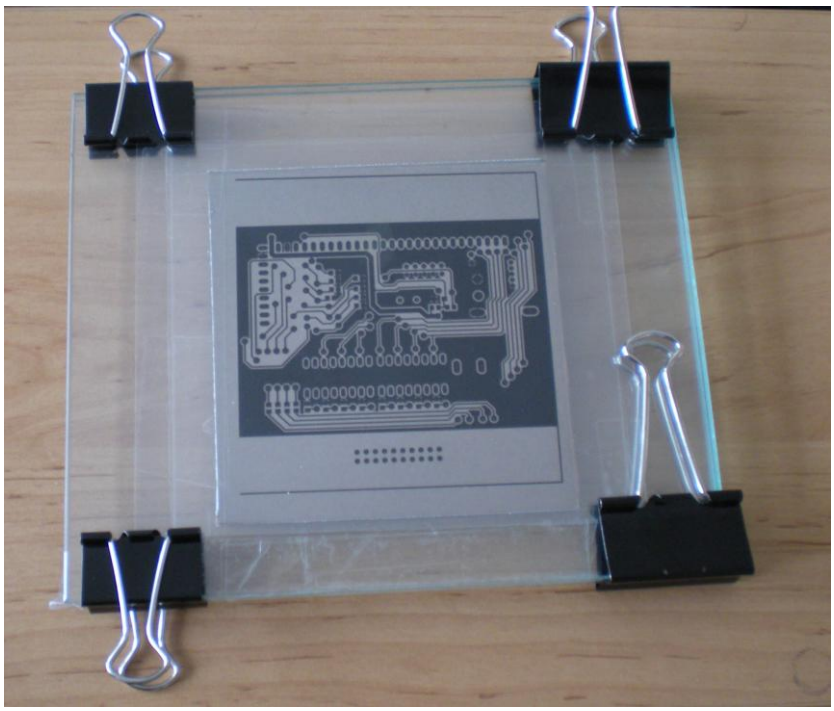


Obr. 19 Předloha osvitu horní DPS

4.3.2 Osvit DPS

Před samotným osvícením cuprextitové desky UV světlem se musí nastavit předloha. U oboustranné DPS se obě předlohy přiloží k sobě a musí mít vůči sobě přesnou polohu. Deska by jinak byla nepoužitelná. Osvědčilo se obě předlohy k sobě slepit na okně, kde se srovnaly podle děr určených pro vrtání. Po slepení obou předloh vznikne kapsa, do které se dá vložit deska pro osvit. Jak bylo uvedeno výše, je důležité, aby potisklé strany fólie byly zevnitř kapsy, jinak by mohlo dojít k rozostření obrazu.

Desku vložíme do vzniklé kapsy a vše umístíme mezi dvě pečlivě očištěné skla. Ty zpevníme svorkami. Přípravek je na obr. 20.



Obr. 20 Přípravek připravený pro osvit

K osvitu byla použita DPS osazená UV LED diodami. Délku osvitu jsme stanovili na 4 minuty na jednu stranu desky. Vzdálenost osvitové jednotky od DPS v přípravku byla přibližně 25 cm.

Po vypnutí UV světla a vyndání desky z přípravku vidíme, že při osvitu došlo k vytvoření negativu na cuprextitové desce. Ve vývojce se z těchto míst odstraní lak a při leptání dojde k odstranění mědi.

4.3.3 Vyvolání DPS

Po osvitu se cuprextitová deska vyjme z přípravku a ponoří do předem připraveného lázně. Roztok, ve kterém vyvolání probíhá, se nazývá vývojka. Jde o rozto hydroxidu sodného. Vývojka se nalije do mělké nádoby dost velké, aby se do ní vlezla osvícená deska.

Pro kvalitnější vyvolání je vhodné pomocí pinzety pohybovat deskou a čistým štětečkem zlehka přetírat osvícené strany. Tak se zajistí lepší proudění a odstraňování odleptaných částic.

Během procesu pak začne postupně vystupovat osvícený obraz. Jakmile dojde k úplnému vykreslení, Vyjmeme desku a opláchneme vodou pro odstranění zbytků vývojky. Celý proces vyvolávání trval přibližně 15 minut.

4.3.4 Leptání

Omytá deska se musí pečlivě osušit před vložením do leptacího roztoku. Chlorid železitý, použitý jako leptadlo, je totiž rozpustný ve vodě a docházelo by tak k znehodnocení roztoku.

V našem případě jsme leptali pouze oboustranné desky, proto jsme si museli dávat pozor, aby při leptání jedné strany nedocházelo příliš ke styku druhé strany s roztokem. Tak by se mohly inkriminovaná místa tzv. podleptána.

Osušená deska byla položena na hladinu, kde plavala. Pokládáme ji tak aby pod ní nevznikaly bubliny, nejdříve tedy desku z jedné strany lehce namočíme a až potom položíme na hladinu. Odleptané částice mědi tak mohou volně odpadávat. Leptání jedné strany trvalo 30 minut. Doba leptání se může prodlužovat v závislosti na četnosti používání roztoku.

Při leptání druhé strany jsme vyleptanou stranu přelepili izolepou, tak byla chráněna před stykem s leptadlem.

Obr. 21 Leptání

4.3.5 Úpravy

Po vyjmutí z leptacího roztoku byla deska omyta. Zbytky laku byly odstraněny očištěním desky acetonem. Dalším krokem bylo vyvrtání otvorů pro součástky a prokovy a následné ořízнutí pomocí pákových nůžek do finálního tvaru. Hrany byly přebroušeny hrubým smirkem, a plocha DPS jemným vodním smirkem pro odstranění nerovností po vrtání. Nakonec se deska pokryla nánosem pájitelného laku. Před pájením se lak nechal schnout 24 hodin.

4.3.6 Osazení

Postup pájení součástek jsme volili podle složitosti procesu. První byl na řadě mikrokontroler ATmega16A, potom oboustranné prokovy tvořené tenkými drátky jako např. zbytky nožiček z LED diod. Poté se napájely SMD, a nakonec ostatní součástky a konektory.

5 PROGRAM MODELU

Po dokončení výroby funkčního modelu automatické pračky bylo nutné udělat poslední a nejdůležitější krok. Bylo zapotřebí vytvořit program ve vývojovém prostředí AVR studio v jazyce C, který by obsahoval všechny potřebné instrukce pro chod modelu.

Napsaný program je nutné následně zkompilovat a pomocí programu AVR ISP programmer a programátoru UniProg-USB nahrát do mikrokontroleru ATmega16A. Mikroprocesor se správně napsaným a nahraným programem potom dokáže vyhodnocovat signály přicházející z PLC a řídit simulace modelu automatické pračky.

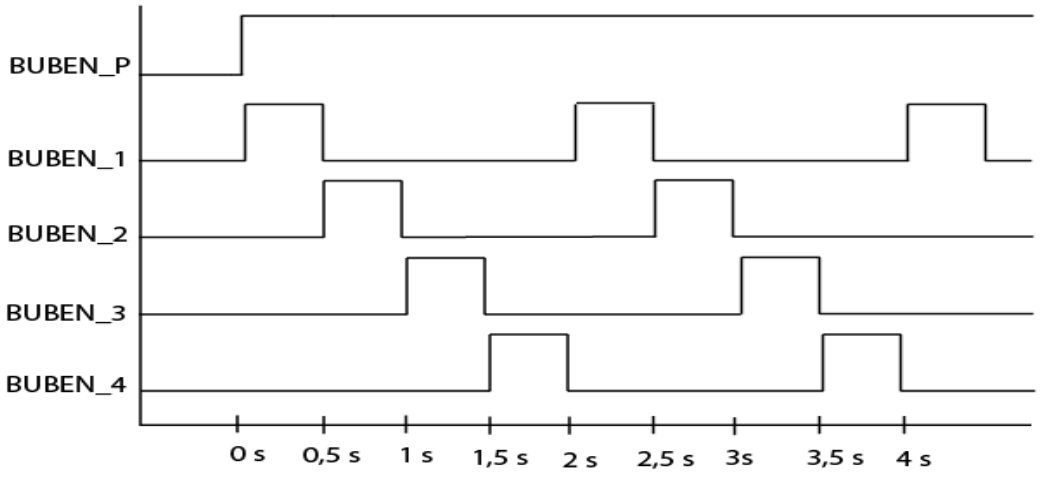
Před samotným programováním bylo nutné odměřit a přesně určit funkce a časy jednotlivých procesů modelu EDUmod. Při této analýze nám vzniklo 5 základních částí programu: inicializační stav, otáčení bubnu, napouštění/vypouštění vody, ohřev a samovolné ochlazování vody.

Největší pozornost byla věnována ohřevu a ochlazování vody v bubnu. Podle stránek výrobce se má model při ohřevu chovat jako soustava 2. řádu, tj. funkce ohřevu má být závislá na aktuálním stavu hladiny. Při měření jsme zjistili, že ohřev při 100% hladině je dvakrát tak dlouhý než při 50% hladině. Pokud ovšem není napuštěna žádná voda, ohřev probíhá stejně jako při 50% hladiny. Toto řešení jsme nahradili lineární funkcí, která sleduje aktuální stav hladiny po 1% a tak zajišťuje rozdílnou dobu ohřevu při zachování stejných časových intervalů ohřevu při 50% a 100% hladině. Pokud se však ohřev spustí při nízké nebo žádné hladině ohřev proběhne velice rychle. To jsme ošetřili novým chybovým hlášením. Pokud přijde povel k ohřevu tak se simulace procesu zastaví a bude nutné model restartovat tlačítkem RESET.

5.1 Měření procesů

5.1.1 Otáčení bubnu

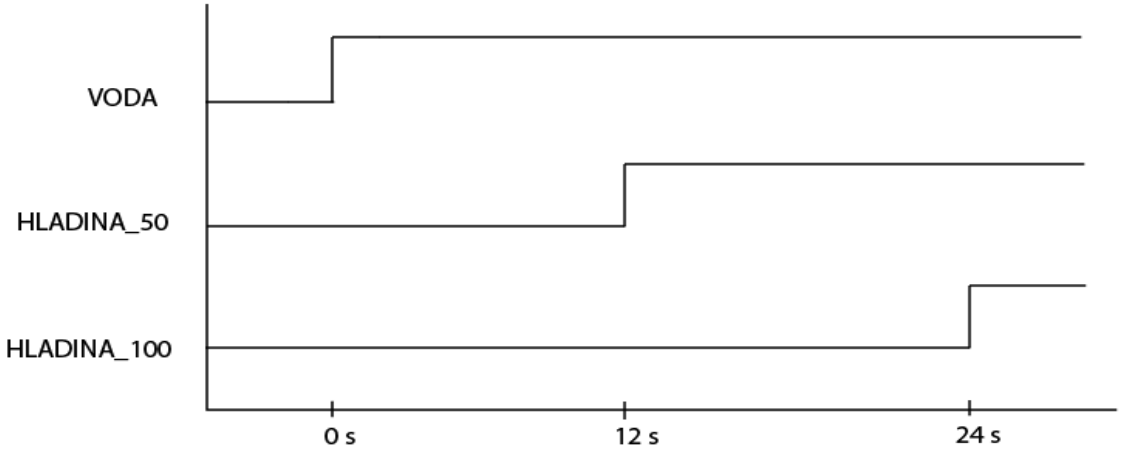
Otáčení bubnu probíhá na 8 LED uspořádaných v kruhu. V našem provedení máme pouze 4 možné stavy. Změření jsme zjistili 0,5 s prodlevu při normálních otáčkách. Průběh otáčení bubnu oběma směry je totožný, pouze po sepnutí pinu OTACKY je prodleva 0,15s. Časový průběh jsme vložili do diagramu na obr. 22.



Obr. 22 Časový diagram simulace otáčení bubnu doprava

5.1.2 Napouštění

Simulace napouštění je vcelku jednoduchý proces, kdy se při příchodu logické 1 na pin VODA začne napouštět voda do bubnu pračky a její hladina je snímána ve dvou stavech (50% a 100%). Výsledky měření jsme uvedli do časového diagramu na obr. 23.

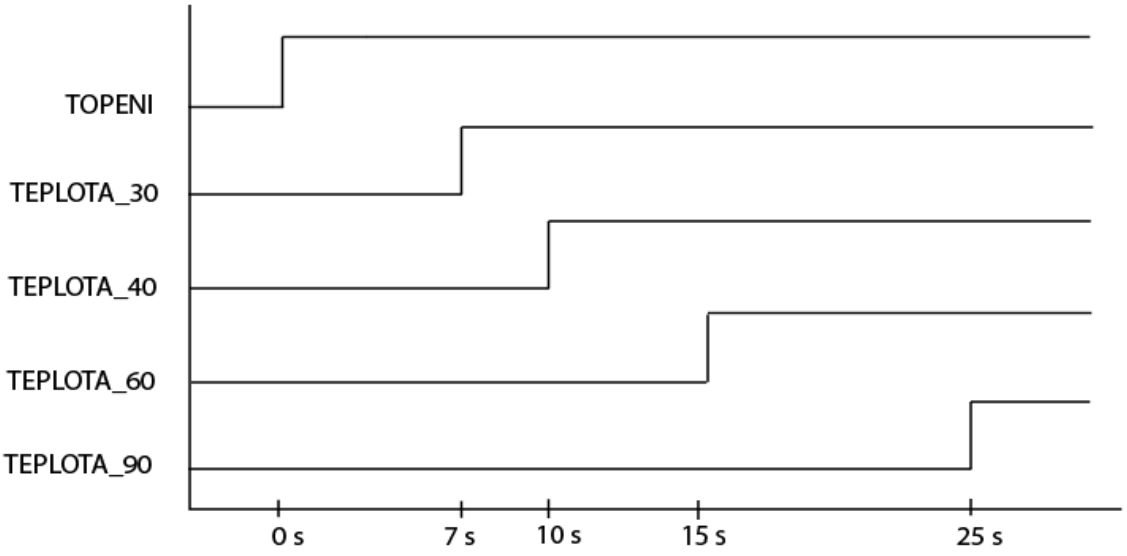


Obr. 23 Diagram napouštění vody

5.1.3 Ohřev

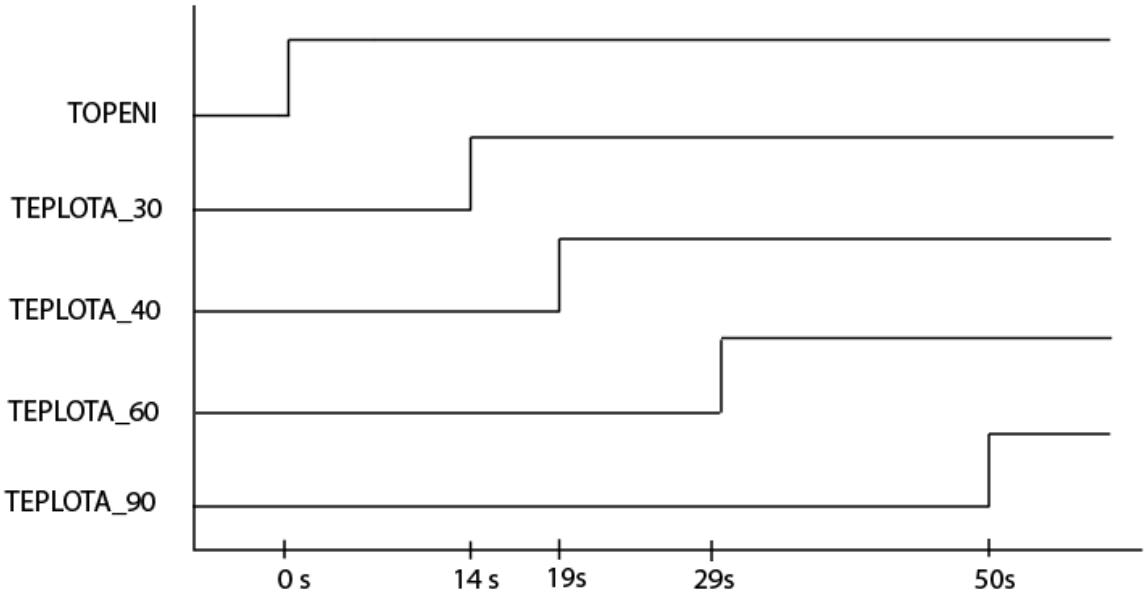
U měření ohřevu se mělo počítat s rozdílným ohřevem v závislosti na naplnění bubnu pračky vodou. Měření jsme prováděli ve třech fázích: v počátečním stavu (hladina = 0%), po sepnutí indikace HLADINA_50 (hladina = 50%) a po sepnutí HLADINA_100 (hladina = 100%).

Z měření vyplynulo, že ohřev vypuštěného bubnu by měl trvat stejně dlouho jako ohřev bubnu napuštěného na 50%. Průběh měření je zobrazen časovým diagramem na obr. 24.



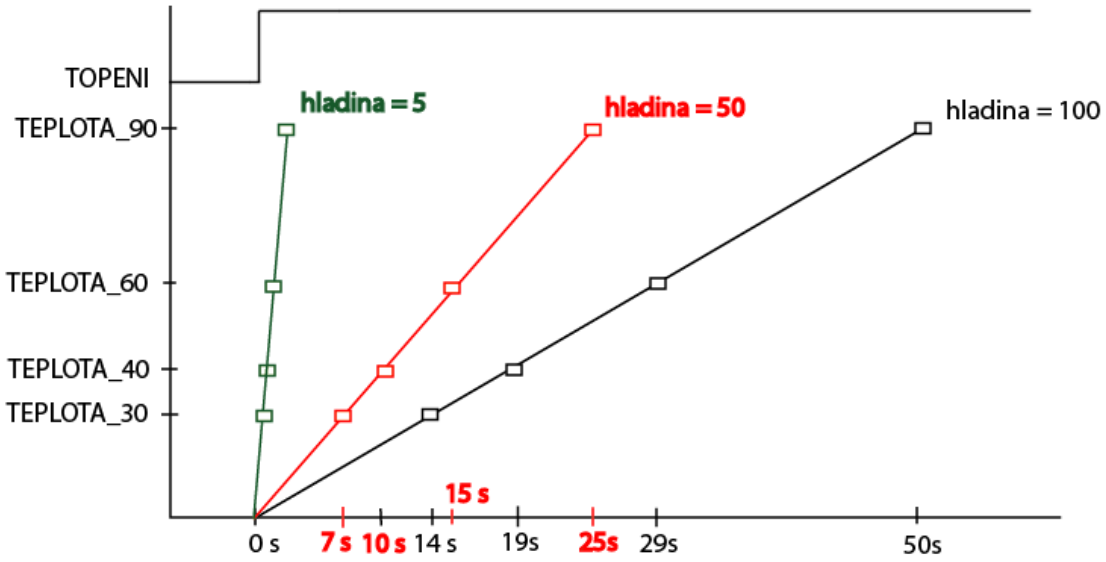
Obr. 24 Průběh ohřevu při hladině 0% a 50%

Při napuštěném bubnu na 100% hladiny byly časy ohřevu přibližně dvojnásobné oproti předchozímu měření, ovšem pouze až po sepnutí indikace HLADINA_100. Měření zobrazeno časovým diagramem na obr. 25.



Obr. 25 Průběh ohřevu při hladině 100%

Po naměření hodnot ohřevu jsme si dali za cíl vytvořit lineární průběh, podle kterého by probíhala simulace ohřevu vody podle aktuálního stavu hladiny, aby tak simulace byla více reálná, než je tomu u modelu EDUmod. Funkce ohřevu by tedy měla odpovídat průběhu, znázorněném na obr. 26.



Obr. 26 Znázornění průběhu ohřevu při různých stavech hladiny

Z obrázku můžeme vidět různé doby simulace ohřevu při rozdílném napuštění vody v bubnu.

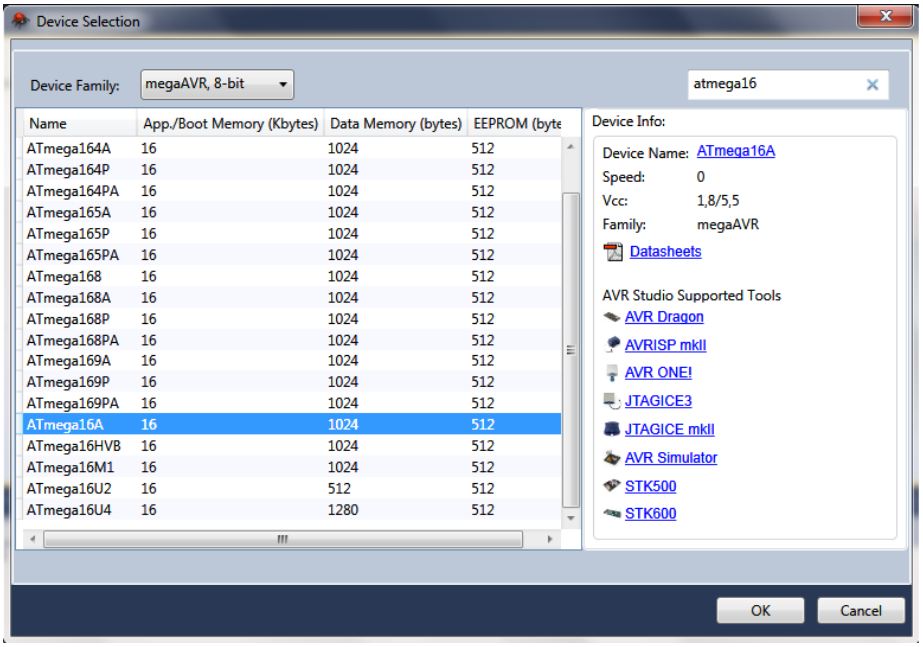
5.2 AVR Studio

AVR Studio je profesionální integrované vývojové prostředí pro vytváření a editaci programových aplikací pro procesory Atmel AVR. Umožňuje nám psát program v jazycích C a Assembler. Pro psaní programu jsme zvolili uživatelsky pohodlnější jazyk C. Takto napsaný program se při sestavení nejprve přeloží do Assembleru a teprve potom se kompiluje do strojového byte kódu, který se nahrává do mikrokontroleru.

Samotný program se vytváří v textovém editoru, který rozeznává a barevně odlišuje různé části kódu, jako jsou instrukce, proměnné, definice atd., což nám zjednodušuje orientaci v napsaném kódu.

Další funkcí je možnost ladění programu přímo v textovém editoru, kde můžeme program trasovat, a přidávat do něj breakpointy.

Při spuštění nového projektu v AVR Studiu se nejdříve vybere typ a jazyk, kterým budeme program psát. V našem případě tedy C projekt s popisem spustitelný (Executable). Potom z výběru mikrokontrolerů firmy Atmel vybereme náš ATmega16A, a potom už se nám zobrazí textový editor, ve kterém píšeme samotný kód programu. Výběr zařízení je na obr. 27.



Obr. 27 Výběr zařízeni mikrokontroleru

5.3 Hlavní části programu

Při psaní programu jsme se museli zamyslet nad principy, které můžeme využít při programování ATmegy v softwaru AVR Studio. Jako nejlepší řešení, s kterým se mohly vyřešit téměř všechny části programu, se nám jevilo použití samostatného časovače mikrokontroleru. Pro naše podmínky jsme si zvolili první z možných časovačů/čítačů, 8-bitový časovač 0 s funkcí přetékání. S jeho pomocí jsme pak byli schopni realizovat části programu, které mají za úkol simulovat procesy otáčení bubnu, napouštění/vypouštění vody a ohřev/chladnutí.

5.3.1 Časovač 0

Časovač je samostatná část mikrokontroleru, která nezávisle na probíhajících instrukcích načítá frekvenci. Frekvence je určena vnitřním zdrojem hodinových impulzů a případnou předděličkou zařazenou do cesty. Tento způsob použití se nejčastěji využívá pro časování pravidelných intervalů.[10]

Popis registrů:

- **TCNT0** - hodnota časovače
- **OCR0** - srovnávací registr
- **TCCR0** - řízení funkce časovače
- **TIMSK** - registr masek přerušování časovačů
- **TIFR** - příznaky časovačů
- **SFIOR** - zvláštní vstupně výstupní registr

Podle výše uvedeného popisu registrů jsme zvolili TCCR0, který slouží k nastavení časovače. Bity registru jsou popsány v tab. 4

Registr TCCR0								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Název	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00
Read/Write	W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Tab. 4 Popis bitů registru TCCR0

Na výběr zdroje hodin a předděličky slouží pouze první tři bity: CS00, CS01 a CS02. V dokumentaci mikrokontroleru ATmega16A najdeme tabulku, podle které vidíme, že obsah registru TCNT0, ze kterého získáváme hodnotu časovače, můžeme inkrementovat vnitřně (přímo, nebo přes nastavenou předděličku), nebo externě. Možnosti nastavení zdroje hodin jsou v tab. 5.

CSO2	CS01	CS00	Popis
0	0	0	Bez zdroje signálu
0	0	1	$clk_{I/O}$ bez předděličky
0	1	0	$clk_{I/O}$ /8 předdělička
0	1	1	$clk_{I/O}$ /64 předdělička
1	0	0	$clk_{I/O}$ /256 předdělička
1	0	1	$clk_{I/O}$/1024 předdělička
1	1	0	Externí zdroj hodin na pin T0 (sestupná hrana)
1	1	1	Externí zdroj hodin na pin T0 (vzestupná hrana)

Tab. 5 Možnosti nastavení zdroje hodinového signálu

Dalším použitým registrem v programu byl TIMSK (Timer/counter nterrupt Mask Register). Pomocí bitu TOIE0 je možné povolit přerušení při přetečení čítače. Bity registru popsány v tab. 6.

Registr TIMSK								
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Název	OCIE	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	-	TOIE0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Tab. 6 Popis bitů registru TIMSK

5.3.1.1 Nastavení časovače

Před samotným použitím časovače v kódu je nutné zvolit a vypočítat hodnoty přerušení, které budeme využívat.

Frekvence hodin je nastavena externím krystalem o hodnotě frekvence 16 MHz. Předděličku si zvolíme /1024. Z těchto hodnot lze vypočítat periodu inkrementace obsahu registru TCNTO pomocí následujícího vzorce:

$$T = \frac{1}{\frac{f}{1024}} = \frac{1}{\frac{16MHz}{1024}} = 64us$$

[5]

Registr se tedy inkrementuje každých 64us. Časovač je 8-bitový, takže přetečení může nastat až po 256 impulzech časovače. Z těchto poznatků poznat snadno sestrojíme vzorec, ze kterého vyplyne, jak často nastane přetečení:

$$P = 2^8 * T = 256 * 64us = 16{,}384ms$$

[6]

Posledním výpočtem zjistíme, kolikrát musí přetečení nastat, aby dosáhlo na určitou časovou hodnotu. Ve vzorovém výpočtu hledáme počet přerušení čítače, když chceme dojít až na 1 vteřinu.

$$N = \frac{1}{P} = \frac{1000ms}{16{,}384ms} \cong 61{,}035 \quad [7]$$

Tento vzorec jsme využívali u každé části programu, která měla být závislá na časové prodlevě.

Posledním krokem byl zápis časovače do kódu programu tak aby povoloval přerušení a využíval námi zvolenou předdělička 1024. Nastavení časovače bylo provedeno podle tab. 5 a zápis v kódu je na obr. 23.

```
TCCR0 |= (1 << CS02) | (1 << CS00);      // preddelicka /1024 (1us)
TIMSK |= (1 << TOIE0);      // prerusení pri pretečení TCNT0
```

Obr. 28 Nastavení časovače 0

5.3.2 Otáčení bubnu

Jak bylo vše nastaveno mohli jsme přejít k samotnému psaní kódu. Jako první jsme řešili problematiku otáčení bubnu. Otáčky se řídí třemi členy, z toho dva řídí směr otáčení (BUBEN_L a BUBEN_P) a poslední mění rychlost otáček na vyšší, což simuluje ždímání prádla (OTACKY).

Nejdříve jsme určili stavy chybového hlášení. Zde se vyskytuje tzv. opravitelná chyba. Pokud přijde příkaz současně otáčet bubnem doleva i doprava. Všechny simulace se zastaví a LED ERR začne blikat. Tuto závadu lze odstranit jednoduše opravením příkazu, a model dál funguje bez nutnosti resetu mikrokontroleru. Realizace je na obr. 29.

```
if(bit_is_set(PINC,0) && bit_is_set(PINC,1))
    {
        ERR_on;
        _delay_ms(500);
        ERR_off;
        _delay_ms(500);
    }
```

Obr. 29 Realizace opravitelné chyby u otáčení bubnu

Signál BUBEN_P je umístěn na pinu 0 portu PC, signál BUBEN_L je na stejném portu na pinu 1. Příkaz ERR_on a ERR_off jsme si nadefinovali na začátku programu. V definici je nastaveno převracení logiky, které je potřebné u všech výstupních členů ATmegy, resp. vstupních členů PLC. Definice je na obr. 30.

```
#define ERR_on  PORTA &= ~(1 << PA7) // zápis log.0 na pin 6 portu
#define ERR_off PORTA |= (1 << PA7) // zápis log. 1 na pin 6 portu
```

Obr. 30 Definice s převrácením logiky

Je tedy obstaráno, aby nenastal případ, když by na model přišel příkaz na točení doleva a doprava zároveň. Teď se podíváme, co se stane, pokud přijde pouze jeden z těchto signálů.

Otáčení bylo realizováno pomocí pomocné proměnné *pozice*. Buben byl vytvořen z 8 LED uspořádaných do kruhu. Vždy dvě protichůdné diody byly spojeny. To nám dává 4 možné pozice při otáčení bubnu. Pomocná proměnná nám nabývala hodnoty od 1 do 4, přičemž každá pozice odpovídala vypnutí předchozího signálu a zapnutí nové dvojice LED signálem. Realizace v kódu je na obr. 31.

```
if(i == 30)
{  // proběhne každých 500ms

    if(bit_is_set(PINC,0) && bit_is_clear(PINC,1)) // točení doprava
        {
            if(pozice==1) {BUBEN4_off;BUBEN1_on; }
            if(pozice==2) {BUBEN1_off;BUBEN2_on; }
            if(pozice==3) {BUBEN2_off;BUBEN3_on; }
            if(pozice==4) {BUBEN3_off;BUBEN4_on; pozice=0;}
            pozice++;
        }
        i=0;
}
i++;
```

Obr. 31 Realizace otáčení bubnu

V pomocné proměnné *i* je uložena hodnota kolikrát musí nastat přetečení, aby se příkaz mohl provést. Při otáčení se poloha pračky mění každých 500ms. Pokud vteřina vyžadovala 61x přerušení u polovičního času volíme 30x.

Při opačném směru otáčení se proměnná *pozice* bude odčítat. Když dojde na konec, nastaví se *pozice* = 5. To aby se nepřeskočila pozice č. 4. Potom se pouze u každého stavu pozice přehodí *on* za *off*, a obráceně.

Při příchodu signálu na pin OTACKY (PIN, 2) je rozdíl pouze v proměnné, obsahující počet přetečení. Ta obsahuje nižší hodnotu počtu přerušení, aby se proměnná *pozice* a s ní i LED přepínali rychleji.

5.3.3 Napouštění/vypouštění vody

U kontroly stavu hladiny jsme definovali novou pomocnou proměnnou *hladina*, která dosahuje hodnot od 1 do 105. Při příchodu signálu na pin VODA (PINC, 4) se začne tato proměnná inkrementovat. Na obr. 32 je provedení v programu.

```
if(bit_is_set(PINC,4)) // napousteni
        {
            if(p == 15)     // proběhne
            {
                hladina++;
                if(hladina == 50)  {HLADINA50_on; }
                if(hladina == 100) {HLADINA100_on; }
                if(hladina == 105) {ERR_on; err = 1;}
                p=0;
            }
        p++;
        }
```

Obr. 32 Realizace napouštění vody

Z modelu EDUmod jsme naměřily doby napouštění na 50% a 100% hladiny (12s a 24s). Jelikož proměnná hladina nabývá hodnoty od 1 do 105, tak stačilo zjistit čas napouštění 1% hladiny (250ms) a pak podle vzorce na počet přerušení dalo zjistit potřebný počet přerušení. Ten byl zadán do proměnné *p*.

Při přetečení hladiny na 105% (*hladina* = 105) přejde model do neopravitelného chybového stavu, ze kterého se lze dostat pouze restartováním modulu tlačítkem umístěným na horní DPS.

Vypouštění probíhá analogicky obráceně. Při příchodu logické 1 na pin CERPADLO (PINC, 5), se proměnná *hladina* dekrementuje a při určených hodnotách se LED vypínají. Je nutné zamezení dekrementace do záporného stavu *hladiny*.

5.3.4 Ohřev a chladnutí

Nejsložitější funkcí modelu automatické pračky je ohřev a samovolné ochlazování vody v bubnu. To z důvodu závislosti na aktuálním stavu hladiny, který se u modelu EDUmod omezuje pouze na dva druhy ohřevu a jeden druh ochlazování. V našem návrhu jsme zvolili variantu, kde lineární rovnice bude sledovat aktuální stav po 1% hladiny a to bude určovat rychlost ohřevu. Byla tedy využita proměnná *hladina* a nadefinovaná nová proměnná *ohřev*, která nabývá hodnot od 1 do 1000.

Z naměřených hodnot jsme zjistili ohřev při 100% hladině na 90°C (t = 50 000ms). Pokud tedy umístíme sepnutí TEPLOTA_90 na *ohřev* = 1000 pak zjistíme, že inkrementace proměnné musí proběhnout každých 5 ms. Dosadíme do vzorce a vyjde nám, že časovač potřebuje 3x přerušit aby jeden krok trval 50 ms.

Pokud tedy přijde logická 1 na pin TOPENI (PINC, 3), začne se po 50ms inkrementovat proměnná *ohřev* a podle její aktuální hodnoty se spínají bity znázorňující teplotu. Realizace ohřevu je na obr. 33.

```
if(bit_is_set(PINC,3)) // zapnutí  TOPENI
        {
            if(a == 3)      // proběhne
            {
                ohrev++;

                if((hladina*3) == ohrev ) {TEPLOTA30_on; }
                if((hladina*4) == ohrev ) {TEPLOTA40_on; }
                if((hladina*6) == ohrev ) {TEPLOTA60_on; }
                if((hladina*10) == ohrev ) {TEPLOTA90_on; }
                if(((hladina*10)+5) == ohrev ) {ERR_on; err = 1;}

                a=0;
            }
        a++;
        }
```

Obr. 33 Realizace ohřevu vody

Do pomocné proměnné *a* opět zapíšeme potřebný počet přerušení pro inkrementaci o jeden *ohřev*. V kódu se dále ověřuje lineární rovnice, která porovnává n-tý násobek proměnné *hladina* s proměnnou *ohřev*. Pokud dojde ke shodě, provede se příkaz sepnutí, který je nadefinován výše v programu.

Poslední řádek se zabývá neopravitelným chybovým stavem. Pokud dojde k přehřátí o 5 jednotek proměnné *ohřev,* rozsvítí se LED ERR a díky proměnné *err* se simulace modelu zastaví. Z tohoto stavu je možné se dostat pouze restartováním modulu tlačítkem RES na horní DPS.

Chladnutí probíhá analogicky převráceně. Při logické 0 na pinu TOPENI se proměnná *ohřev* odečítá a podle podobných rovnic se porovnává stav a postupně se rozepínají piny představující stav teploty. Pro chladnutí jsme zvolili delší časovou prodlevu.

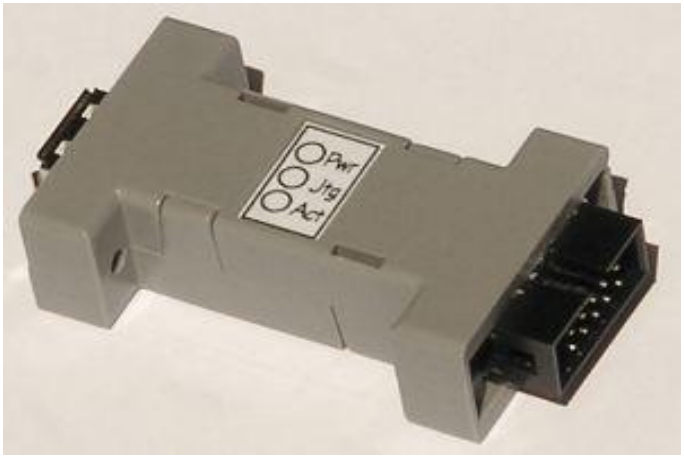
Pokud by při sepnutí bitu TOPENI, byla hodnota proměnné *hladina* nízká, nebo nulová, ohřev by proběhl velice rychle, což se nám pro simulaci nehodí. V tomto případě jsme museli zavést nový neopravitelný chybový stav. Ten nám zajišťuje, že se nebude ohřívat dříve, než bude voda napuštěna minimálně na 5% hladiny. V jiném případě se model zasekne a bude jej nutno restartovat. Realizace chybového stavu je na obr. 34.

```
if(bit_is_set(PINC,3) && (hladina <= 5)) {ERR_on; err = 1;}
```

Obr. 34 Realizace chybového stavu při nízkém stavu hladiny

5.4 Programátor UniProg-USB

Jde o univerzální vysokorychlostní programátor AVR využívající ISP rozhraní, od firmy PK Design. Umožňuje i programování a ladění programu přes JTAG rozhraní z AVR Studia. Tento programátor je součástí vybavení laboratoře A1/731a



Obr. 35 Programátor UniProg

Vlastnosti:

- Jednoduché připojení k osobnímu počítači přes USB port.
- Plnohodnotné galvanické oddělení.
- Napájení 3.3V-5.0V (z připojeného modulu).
- Programovací konektor typu MLW10
- Automatické spuštění ISP/JTAG firmware na základě detekce typu připojeného kabelu.[8]

5.5 AVR ISP Programmer

Výrobek formy PK Design umožňuje programování a čtení FLASH EEPROM paměti mikrokontrolerů řady AVR vstupními soubory HEX, které jsou standardně generovány vývojovým prostředím AVR Studio. Našim potřebám se také hodí možnost programování FUSE bitů. Program se běžně využívá právě s programátorem UniProg-USB, který jsme použili. [12]

Program automaticky rozpoznává připojení programátoru a typ připojeného mikrokontroleru. Sám si pak zvolí nastavení potřebných parametru, jako je velikost FLASH paměti, EEPROM paměti a typy FUSE bitů. Kromě toho vypíše informace o obvodu.[12]

Čtení a zápis jsou během provádění příkazu kontrolovány sledováním odezvy z mikrokontroleru tak, že každý poslaný bit před ISP programátor se zpětně porovnává s vyslaným.

Před nahráním programu bylo nutné správně nastavit FUSE bits (pojistky) aby program v mikrokontroleru fungoval jak má. Nastavení pojistek se může různit podle

použitého obvodu, podle použití externího/interního krystalu a jeho velikosti, spotřeby atd. V našem případě jsme pojistky nastavili podle dokumentace mikrokontroleru ATmega16A. Nastavení FUSE bits je na obr. 36.

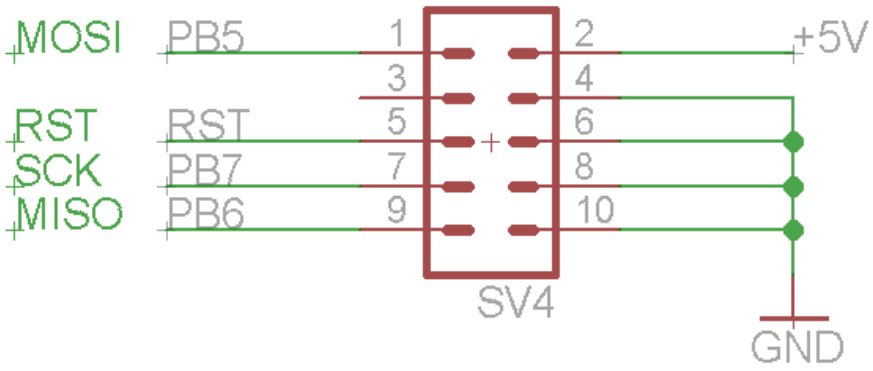
Obr. 36 Nastavení pojistek FUSE bits

5.6 Protokol ISP

Protokol ISP využívají sériové programátory určené pro práci s mikrokontrolery firmy Atmel. Jejich výhodou je cena a velmi jednoduchá konstrukce. ISP umožňuje pouze 6. vodičové připojení mezi programátorem a procesorem. Konektor se skládá z dvou vodičů s přivedeným napájecím napětím, ovládání pinu RESET procesoru, vodiče pro zápis MOSI a čtení MISO obsahu interní paměti FLASH a EEPROM. Poslední vodič je pro synchronizaci přenosu SCK.[13]

Zabrané vodiče na DPS jsou však na dále využitelné, přestože budeme mikrokontroler dále programovat přímo v aplikaci. Z této vlastnosti je i pojmenování protokolu ISP: *In System Programing*. Při ladění programu bylo nutné napájet DPS aby mohl probíhat přenos dat. [13]

Konektor pro ISP programování byl realizován 10 pinovým konektorem MLW10G. Jeho zapojení je na obr. 37.



Obr. 37 Konektor na programování ISP

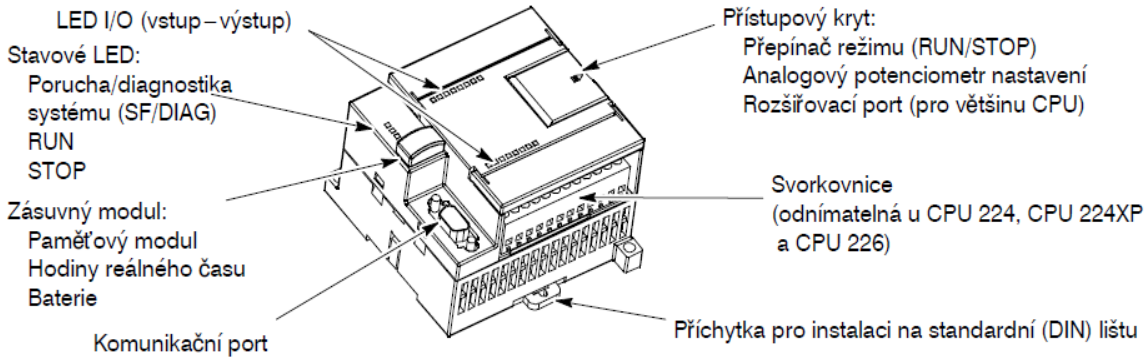
6 ŘÍZENÍ MODELU Z PLC

V této kapitole se věnujeme ověření funkčnosti modelu automatické pračky jejím řízením s pomocí PLC. Vše co bylo použito k řízení je součástí výbavy laboratoře A1/731a ve které celá práce vznikala. K tomuto účelu zde byly PLC Simatic S7-200 od firmy Siemens, spolu s vývojovým prostředím STEP 7 nainstalovaným na počítačích v učebně.

6.1 Simatic S7-200

Jde o celou řadu malých programovatelných logických automatů, které jsou určeny pro řízení spíše jednoduchých aplikací v automatizaci. Jejich výhodou je kompaktní design, nízké náklady a výkonné instrukce. To vše zajišťuje jednoduchost automatu při zachování vysokého výkonu.[15]

PLC je ovládáno uživatelským programem, který může obsahovat různé instrukce přes čítače a časovače, po Booleovskou logiku, matematické funkce atd. Automat kontroluje svoje vstupní členy a podle nahraného programu mění svoje výstupy. Popis automatu je na obr. 38.



Obr. 38 Popis automatu Simatic S7-200

Automatů S7-200 je několik druhů, rozdílem je jednotka CPU. Použitý automat obsahoval CPU 224XP.

Vlastnosti CPU 224XP:

- Max. paměť programu – 12288 B
- Paměť pro data – 10240 B
- Zálohování dat – 100 hodin
- Integrované I/O
  - Digitální – 14 vst./10 výst.
  - Analogové – 2 vst./1 výst
- Vysokorychlostní čítače
  - Jednofázové
    - 4 při 30Hz
    - 2 při 200 Hz
  - Dvoufázové
    - 3 při 20 Hz
    - 1 při 100 Hz

- Pulzní výstupy (DC) – 2 při 100 kHz
- 2x analogový potenciometr
- Integrované hodiny reálného času
- 2x komunikační port RS-485
- Rychlost booleovských instrukcí – 0,22 us na instrukci

6.2 STEP 7

STEP 7 je základní software pro konfiguraci a programování řídicích systémů SIMATIC. Obsahuje výkonné nástroje a funkce pro řadu úloh spojených s automatizačními projekty. Nabízí uživatelsky příjemný způsob práce ve všech fázích vývoje projektu, jakými jsou obvykle konfigurace a parametrizování hardwaru, definování komunikace, programování, testování a oživování projektu, servis, správa dokumentace a archivování, provozní a diagnostické funkce.[16]

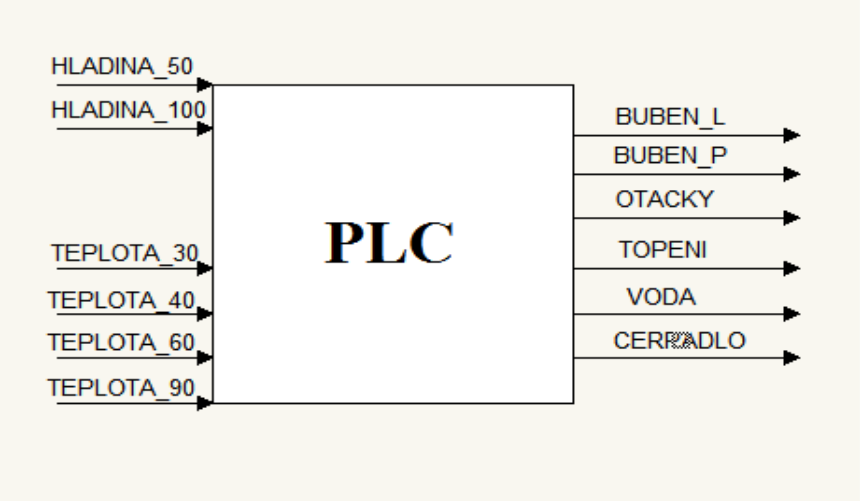
Vlastnosti:

- SIMATIC Manager – integrovaná správa nástrojů a dat projektu
- Programový editor – tvorba uživatelských programů
- Editor symboliky – správa globálních proměnných
- Diagnostika hardwaru – přehled o stavu systému
- NetPro – nastavení datových spojení přes MPI nebo PROFIBUS

6.3 Program na ověření funkčnosti

Pro ověření modulu jsme vzali program vytvořený v předmětu Programovatelné automaty (VPL). Program je finální verzí vytvořenou po 2 hodinách výuky, a pro naše účely je perfektní, protože model, který jsme vytvořili, bude sloužit právě k účelům tohoto předmětu.

Všechny programy, které budou vytvářeny pro modul automatické pračky, se musí řídit podle schématu zobrazeného na obr. 39., který zobrazuje, které členy budeme slevovat a používat a do kterých budeme zapisovat.



Obr. 39 Schéma pro navrhování programů

6.3.1 Zadání

Po stisku tlačítka START se začne napouštět voda do bubnu pračky. Po naplnění nádrže na 50% se spustí ohřev. Po naplnění bubnu vodou na 100% vypnout přítok vody. Po ohřátí vody na 60°C vypnout ohřev a zároveň zapnout otáčení bubnu pračky střídavě 5 s doleva a 5 s doprava s 1 s přestávkou mezi přepnutím směru. Tento stav bude 25 s, a poté se začne buben točit pouze doprava a zároveň se začne odčerpávat voda z bubnu pračky. Po 20 s se začne otáčkami odstřeďovat po dobu 15s. potom se pračka uvede do klidu.

6.3.2 Provedení

V programu STEP 7 jsme si nejdříve zvolili jazyk programování na STL příkazy.

Program je rozdělený na Hlavní program (Main) a na jednotlivé kroky (Podprogramy), konkrétně 6 pro větší přehlednost. Načítání jednotlivých kroků probíhá přes proměnnou KROK, která nabývá hodnot 0-5. Z této proměnné se načte její hodnota a podle toho se zavolá příslušný podprogram. Hlavní program Main je tedy jenom „křižovatkou“, která odkazuje na jednotlivé kroky. Příklad volání podprogramů je na obr. 40.

```
LD      SM0.1
MOVB    0, KROK:MB0

LDB=    0, KROK:MB0
CALL    SBR_0:SBR0
```

Obr. 40 Inicializace a krokování programu

Bit SM0.1 se využívá pro počáteční inicializaci, bude tedy na začátku každého podprogramu.

V nultém kroku, podprogramu, se resetují všechny proměnné a časovače. To z důvodu kdyby v programu byla chyba a program se dostal do tohoto stavu se zapnutými bity.

První krok programu spíná napouštění vody. Při oznámení “senzoru“, že bylo naplněno na 50% (HLADINA50) sepne topení a při naplnění na 100% (HLADINA100) vypne vodu a přenastaví proměnnou KROK na hodnotu 2. Program se tak vrátí do hlavního programu MAIN, kde bude přesměrován na druhá krok programu. Kód prvního podprogramu je na obr. 41.

```
LD      SM0.0
S       VODA:Q0.4, 1

LD      HLADINA50:I0.4
S       TOPENI:Q0.3, 1

LD      HLADINA100:I0.5
R       VODA:Q0.4, 1
MOVB    2, KROK:MB0
```

Obr. 41 Kód 1. kroku

V druhém kroku se pouze čeká na ohřev vody na 60°C, jakmile je této hodnoty dosaženo bit TOPENI se resetuje a přejde se stejným způsobem jako minule na třetí krok.

```
LD      SM0.0
S       TOPENI:Q0.3, 1

LD      TEPLOTA60:I0.3
R       TOPENI:Q0.3, 1
MOVB    3, KROK:MB0
```

Obr. 42 Kód 2. kroku

V třetím kroku sepneme na začátku předem definovaný časovač T40 nastavený na hodnotu *Tpra* = 25 s. Samotné střídavé otáčení je realizováno zobrazením nabývajících bitů časovače T40 do paměťového bytu MW10. Z toho se pak čtou stavy 5. a 6. bitu, které se mění přibližně každých 5 vteřin. Po doběhu časovače se oba druhy otáček resetují a program je poslán do čtvrtého kroku. Kód třetího kroku je na obr. 43.

```
LD      SM0.0
TON     T40, Tpra:VW2

LD      SM0.0
MOVW    T40, MW10

LDN     M11.5
R       BUBENVP:Q0.0, 1
R       BUBENVL:Q0.1, 1

LD      M11.5
AN      M11.6
S       BUBENVP:Q0.0, 1
R       BUBENVL:Q0.1, 1

LD      M11.5
A       M11.6
R       BUBENVP:Q0.0, 1
S       BUBENVL:Q0.1, 1

LD      T40
R       BUBENVP:Q0.0, 1
R       BUBENVL:Q0.1, 1
MOVB    4, KROK:MB0
```

Obr. 43 Kód 3. kroku

Čtvrtý krok začne odčerpávat vodu spolu se zapnutím otáček bubnu doprava. Časovač T41 je nastaven na 20 s. Poté se oba procesy zastaví a přejde se k poslednímu kroku. Kód je na obr. 44.

```
LD      SM0.0
S       CERPADLO:Q0.5, 1
S       BUBENVP:Q0.0, 1
TON     T41, Tvyp:VW4

LD      T41
R       CERPADLO:Q0.5, 1
R       BUBENVP:Q0.0, 1
MOVB    5, KROK:MB0
```

Obr. 44 Kód 4. Kroku

V posledním podprogramu se opět zapne odčerpávání, k tomu se přidá sepnutí bitu OTACKY. Pračka tehdy simuluje rychlé otáčky bubnu, které představují ždímání. Je tu nastaven časovač T42 na 15 s. Potom se oba procesy vypnou a proměnná KROK je nastavena na počáteční hodnotu. Program potom bude fungovat od znova.

7 ZÁVĚR

Ke splnění cíle jsme se museli seznámit s modelem automatické pračky EDU-mod, a podle získaných poznatků sestrojit a programově realizovat obdobný model. Funkčnost modelu měla být ověřena naprogramováním PLC.

V druhé kapitole jsou popsány vlastnosti vybraného modelu EDU-mod, který se nachází se v laboratoři A1/731a, pro potřeby výuky.

Třetí kapitola se zabývá navrženým hardwarem nového modulu automatické pračky. Popisuje použitý mikrokontroler ATmega16A a je zde uvedeno řešení napájení modelu, obousměrná komunikace modelu s PLC i zobrazování stavů.

Ve čtvrté kapitole je podrobně popsán postup výroby DPS, ze kterých se model skládá, od návrhu až po samotnou realizaci.

Pátá kapitola se zabývá způsoby, jakým byl programován procesor modelu. Popisuje vývojové prostředí, hlavní části vytvořeného řídícího programu a programátor použitý k nahrání do mikrokontroleru.

Poslední kapitola popisuje programovatelný automat Siemens S7-200 a jeho vývojové prostředí STEP7. Dále popisuje program použitý ke kontrole funkčnosti modelu.

Cíle práce se podařilo splnit. Dále byla navržena univerzální DPS, která může být použita při navrhování nových modelu.

SEZNAM POUŽITÉ LITERATURY

[1] KOHOUT. *Modely EDU-mod* [online]. 2008 [cit. 2012-05-22]. Dostupné z:http://www.edumat.cz/produkty.php?produkt=edumod

[2] LM2576 : Datasheet [online]. 2004 [cit. 2012-05-22]. Dostupné z: http://www.gme.cz/dokumentace/330/330-113/dsh.330-113.1.pdf

[3] ATmega16A : Datasheet [online]. 2009 [cit. 2012-05-22]. Dostupné z: http: www.gme.cz/dokumentace/958/958-176/dsh.958-176.1.pdf

[4] TTL (logika). In Wikipedia : the free encyclopedia [online]. 22. 5. 2012 [cit. 2012-05-22]. Dostupné z: http://cs.wikipedia.org/wiki/TTL_(logika)

[5] *Eagle online* [online]. 2003 [cit. 2012-05-22]. Dostupné z: http://www.eagle.cz/

[6] Eagle - návod. *PaJa - Elektronika* [online]. 2002 [cit. 2012-05-22]. Dostupné z: http://paja-trb.unas.cz/elektronika/eagle/eagle_navod.html

[7] Pertinax (materál). In Wikipedia : the free encyclopedia [online].2012 [cit. 2012-05- 22]. Dostupné z: http://cs.wikipedia.org/wiki/Pertinax_(materi%C3%A1l

[8] Výroba plošných spojů fotocestou. *Mlab.cz* [online]. 2011 [cit. 2012-05-22]. Dostupné z: http://www.mlab.cz/Articles/HowTo/How_to_make_PCB/DOC/HTML/How_to_make_PCB.cs.html

[9] Vývojové prostředí AVR Studio 5. *Bastli.cekuj.net* [online]. 2012, č. 6 [cit. 2012-05-22]. Dostupné z: http://www.bastli.cekuj.net/2012/02/18/6-vyvojove-prostredi-avr-studio-5-2/

[10] Čítače a časovače. *Kvetakov.net* [online]. 2006 [cit. 2012-05-22]. Dostupné z: http://www.kvetakov.net/clanky/avr/42-citace-a-casovace.html

[11] Programujeme AVR v jazyku C - 4. časť. *Svetelektro.com* [online]. 2012 [cit. 2012-05-18]. Dostupné z: http://svetelektro.com/clanky/programujeme-avr-v-jazyku-c-4-cast-453.html

[12] HARDWARE: PROGRAMÁTORY. *PK-design* [online]. [cit. 2012-05-22]. Dostupné z: http://pk-design.net/HtmlCz/ProgCables.html

[13] SOFTWARE: UTILITY. *PK Design* [online]. [cit. 2012-05-22]. Dostupné z: http://pk-design.net/HtmlCz/SoftUtilities.html#AtmelAVR3

[14] Začínáme s AVR. HW.CZ. *Hardwarové prostředky* [online]. 2000 [cit. 2012-05-22]. Dostupné z: http://avr.hw.cz/programy/startujeme.html

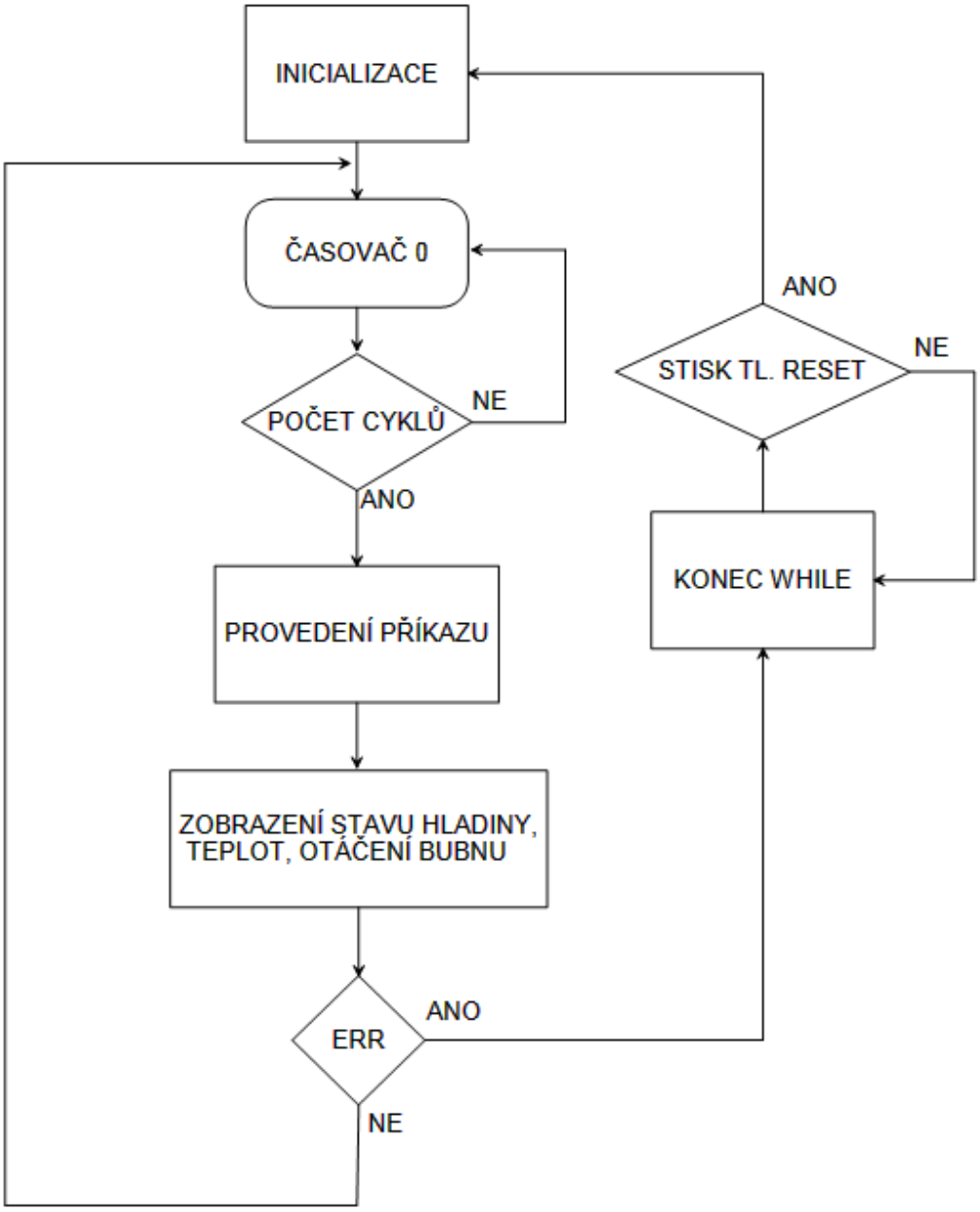
[15] Řídicí systém Simatic S7-200. SIEMENS. *Siemens.cz* [online]. 2012 [cit. 2012-05-22]. Dostupné z: http://www1.siemens.cz/ad/current/index.php?ctxnh=86f90bfae0&ctxp=home

[16] STEP 7. SIEMENS. *Siemens.cz* [online]. 2012 [cit. 2012-05-22]. Dostupné z: http://www1.siemens.cz/ad/current/index.php?ctxnh=3a01fb9720&ctxp=home

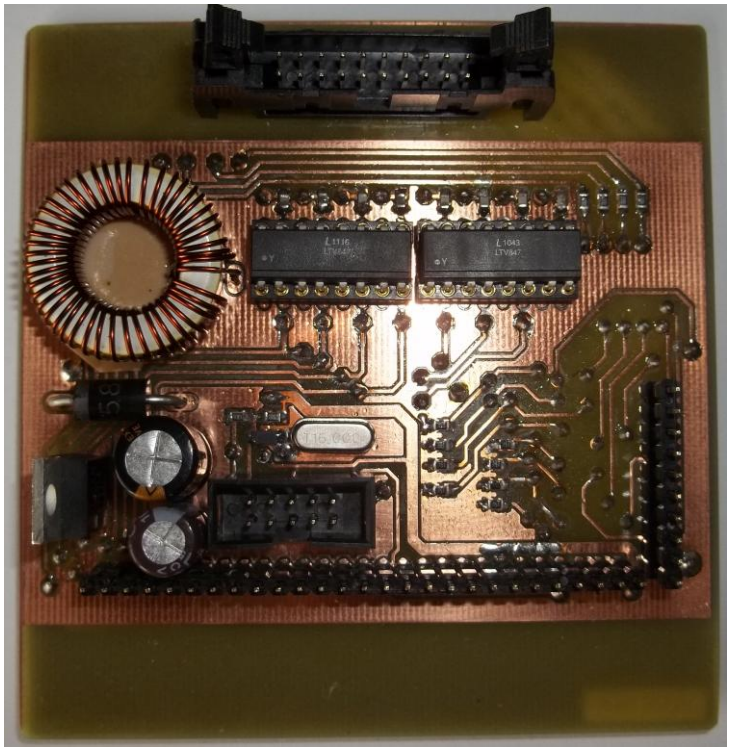
[17] SIEMENS. *SIMATIC*: *Programovatelný automat S7-200 Systémový manuál* [online]. 2004. [cit. 2012-05-22]. Dostupné z: http://www1.siemens.cz/ad/current/content/data_files/automatizacni_systemy /mikrosystemy/simatic_s7200/manual_s7_200_2004_cz.pdf

SEZNAM PŘÍLOH

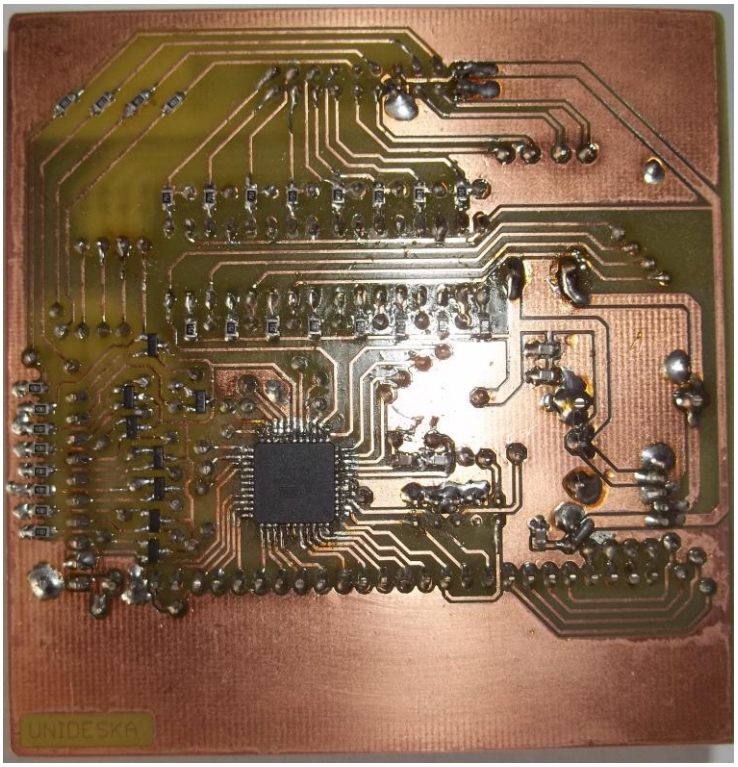
Příloha 1:


INICIALIZACE
ČASOVAČ 0
POČET CYKLŮ
NE
ANO
PROVEDENÍ PŘÍKAZU
ZOBRAZENÍ STAVU HLADINY, TEPLOT, OTÁČENÍ BUBNU
ERR
ANO
NE
STISK TL. RESET
ANO
NE
KONEC WHILE

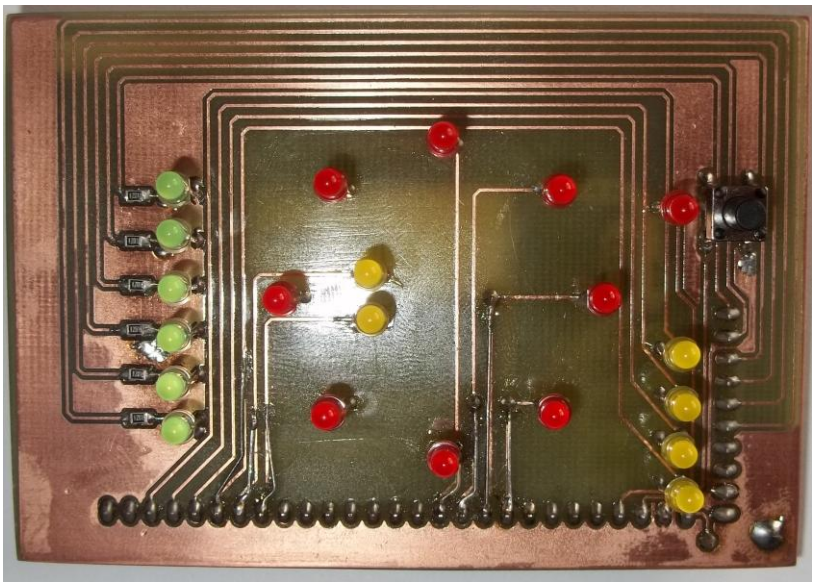
Příloha 1:



Příloha 2:



Příloha 3:



Příloha 4: